Actes du XXXII^{ème} Congrès INFORSID

INFormatique des **OR**ganisation et **S**ystèmes d'**I**nformation de **D**écision

20-23 mai 2014

Lyon

ISBN : 2-906855-29-4
Textes réunis par : Vincent Barrellon, Samuel Gesche et Előd Egyed-Zsigmond

Préface

Je suis heureuse de vous présenter les actes de la 32ème édition du congrès INFORSID qui nous amène cette année à Lyon – la Ville des Lumières.

Créé en 1982, le congrès INFORSID s'est progressivement imposé comme un événement majeur de la communauté francophone des systèmes d'information. Il offre un lieu d'échanges pour les chercheurs, doctorants et industriels sur les recherches et développements liés à l'ingénierie et à la gouvernance des systèmes d'information ainsi qu'aux problématiques connexes émanant d'autres communautés. Il ouvre les portes à la présentation des méthodes et technologies émergentes qui facilitent l'innovation et la création des nouvelles opportunités d'affaires.

Cette année, INFORSID a reçu 70 soumissions de provenances diverses (France, Tunisie, Algérie, Canada, Portugal, Belgique, Cameroun, Mauritanie, Djibouti et Arabie Saoudite). Leur évaluation s'est déroulée en deux phases. Dans un premier temps, chaque article a été examiné par trois membres du Comité de Programme (CP). Cette phase a été suivie par une méta-évaluation supervisée par les membres du Conseil de Programme (CoP) qui ont dirigé les discussions sur les soumissions afin de résoudre d'éventuels conflits d'évaluation. Le travail des méta-évaluateurs a considérablement facilité la prise de décision lors de la réunion de la sélection finale qui, pour la première fois, a été organisée sous forme d'une web conférence. En tant qu'éditeur de ces actes, je tiens à exprimer ma gratitude aux membres des deux comités et aux examinateurs supplémentaires pour leur précieux travail dans la sélection des articles qui a permis de proposer un programme scientifique de qualité.

Le programme du congrès comporte 23 articles regroupés en 9 sessions thématiques, notamment :

– impact des *-data sur les systèmes d'information,

– ingénierie des exigences : modélisation, vérification et traçabilité,

– manipulation, visualisation et exploitation de modèles complexes,

– systèmes d'information pour l'environnement,

– services,

– évaluation des systèmes d'information,

– gestion des données multimédias en environnement mobile,

– processus : traces, fouilles et modélisation,

– ingénierie des documents et des connaissances.

Les 6 premières sessions de cette liste sont complétées par des ateliers associés qui ont pris des formes différentes : tables rondes, présentations d'articles courts, retours d'expérience des invités industriels, etc.

Le programme du congrès propose également deux conférences invitées présentées par des professeurs éminents : Barbara Weber de l'Université de Innsbruck (Autriche) et Florence Sèdes de l'IRIT à Toulouse, sur les thèmes « The Process of Process Modeling » et « La donnée est-elle soluble dans la mobiquité ? » respectivement.

Enfin, le congrès INFORSID 2014 organise de nouveau le Forum Jeunes Chercheurs qui vise à aider les doctorants en 1ère et 2ème année à élargir leur champ des connaissances et à établir des contacts avec des équipes travaillant sur les domaines similaires ou connexes. Cette année, le forum est présidé par Guillaume Cabanac qui a sélectionné 17 jeunes chercheurs pour présenter leur problématique de recherche lors d'une session dédiée et les a accompagné dans la finalisation de leurs articles publiés dans les actes électroniques. Je remercie chaleureusement Guillaume pour son dévouement et son efficacité dans l'organisation de ce forum.

Quatre institutions Lyonnaises ont réuni leurs efforts pour accueillir INFORSID 2014 ; il s'agit de l'INSA, l'Université Claude Bernard Lyon 1, l'Université Lumière Lyon 2 et l'Université Jean Moulin Lyon 3. J'aimerais remercier le Comité d'Organisation, dirigé par le président Omar Boussaïd et le vice-président Előd Egyed-Zsigmond, pour avoir accepté la lourde charge d'organisation de cet événement et le chaleureux accueil qu'ils nous ont réservé.

Pour conclure, j'aimerais étendre mes remerciements à tous les auteurs pour leur confiance et leurs contributions, les porteurs des sessions thématiques pour leurs propositions, les présidents des sessions pour le bon déroulement du congrès et tous les participants, universitaires et industriels, pour leur participation active au congrès INFORSID 2014.

Enfin, je remercie chaleureusement le bureau de l'association INFORSID pour son soutien continu dans la préparation du programme et la promotion du congrès.

Je vous souhaite à toutes et à tous un excellent congrès, riche en échanges scientifiques et plein de grands moments de convivialité.

Jolita Ralyté

Présidente du Comité de Programme

Conseil du comité de programme

Le processus de sélection des articles publiés dans les Actes du Congrès est, depuis deux ans, organisé à deux niveaux avec un Conseil du Comité de Programme (CoP) additionnellement au Comité de Programme (CP) habituel. Les membres du CoP participent aux évaluations des articles et, en outre, organisent une méta-évaluation d'un pool d'articles qui leur sont affectés. La méta-évaluation consiste à organiser les discussions entre lecteurs de chacun des articles du pool afin de résoudre les conflits d'évaluation et d'aboutir, dans la mesure du possible, à un consensus. Les membres du CoP rédigent, à la fin du cycle de discussions, une brève évaluation de synthèse pour chacun des articles de leur pool d'articles. Seuls les membres du CoP participent à la réunion de sélection finale.

Jean-Michel BRUEL	IRIT, Toulouse
Sylvie CALABRETTO	LIRIS, INSA Lyon
Max CHEVALIER	IRIT, Université de Toulouse – Paul Sabatier
Thierry DELOT	Université de Valenciennes
Agnès FRONT	LIG, Université de Grenoble
Charlotte HUG	CRI, Université Paris 1 Panthéon-Sorbonne
Rushed KANAWATI	LIPN, Université Paris Nord
Olivier LE GOAER	LIUPPA, Université de Pau
Thérèse LIBOUREL	UMR Espace-Dev – UM2, Montpellier
Thomas POLACSEK	ONERA, Toulouse
Philippe RAMADOUR	LSIS, Université Aix-Marseille
Camille SALINESI	CRI, Université Paris 1 Panthéon-Sorbonne
Samira SI-SAID CHERFI	CNAM, Paris

Comité de programme

Jolita RALYTÉ (présidente)	ISS, Université de Genève
Zaïa ALIMAZIGHI	Université Houari Boumedienne, Algérie
Saïd ASSAR	TELECOM & Management Sud Paris
Henri BASSON	LISIC, Université du Littoral Côte d'Opale
Remi BASTIDE	IRIT, ISIS Engineer School, Toulouse
Zohra BELLAHSENE	LIRMM, Université de Montpellier 2
Fadila BENTAYEB	ERIC, Université Lyon 2
David BIHANIC	CALHISTE, Université de Valenciennes
Julien BLANCHARD	LINA, Université de Nantes, Polytech Nantes
Mireille BLAY FORMARINO	I3S, Université Nice Sophia Antipolis
Mourad BOUNEFFA	Université Lille Nord de France (Littoral)
Corine CAUVET	LSIS, Université Paul Cezanne, Marseille
François CHAROY	LORIA, Université de Lorraine, Nancy
Stéphanie CHOLLET	ICIS, INP Grenoble
Vincent COUTURIER	LISTIC, Polythech Savoie, Annecy
Rébecca DENECKERE	CRI, Université Paris 1 Panthéon-Sorbonne
Marlon DUMAS	Université de Tartu, Estonie
Thibault ESTIER	ISI, Université de Lausanne, Suisse
Anne ETIEN	LILF, Polytech Lille
Marie-Christine FAUVET	LIG, Université de Grenoble
Christophe FELTUS	CRP Henri Tudor, Luxembourg
Stéphane FRÉNOT	CITI, INSA Lyon
Jean-Pierre GIRAUDIN	LIG, Université de Grenoble
Christophe GNAHO	Université Paris Est
Claude GODART	LORIA, Université de Lorraine

Nouria HARBI	ERIC, Université Lyon 2
Naoufel KRAEIM	RIADI, ISI de Tunis, Tunisie
Sébastien LABORIE	LIUPPA, Université de Pau et des Pays de l'Adour
Benédicte LE GRAND	CRI, Université de Paris 1 Panthéon-Sorbonne
Xavier LE PALLEC	LIFL, Université Lille 1
Lynda Tamine LECHANI	IRIT, Université Paul Sabatier, Toulouse
Eric LECLERCQ	Le2i, Université de Bourgogne, Dijon
Philippe LOPISTEGUY	LIUPP, IUT de Bayonne
José MARTINEZ	LINA, Polytech Nantes
Alain MILLE	LIRIS, Université Claude Bernard Lyon 1
André MIRALLES	UMR TETIS – IRSTEA, Montpellier
Isabelle MIRBEL	I3S, Université de Nice
Káthia OLIVEIRA	LAMIH, Université de Valenciennes
François PINET	UR TSCF, IRSTEA, Clermont-Ferrand
Christophe PONSARD	CETIC, Charleroi, Belgique
Camille ROSENTHAL-SABROUX	LAMSADE, Université Paris-Dauphine
Florence SEDES	IRIT, Université Paul Sabatier, Toulouse
Farida SEMMAK	Université Paris Est
Abdelhak-Djamel SERIAI	LIRMM, Université de Montpellier
Chantal SOULÉ-DUPUY	IRIT, Université Toulouse 1 Capitole
Carine SOUVEYET	CRI, Université de Paris 1 Panthéon-Sorbonne
Christine VERDIER	LIG, Université de Grenoble
Isabelle WATTIAU	CNAM Paris, CEDRIC

Relecteurs additionnels

Idir Amine Amarouche	Université Houari Boumedienne, Algérie
Catherine Berrut	LIG, Université de Grenoble
Vincent Blondeau,	LILF, Polytech Lille
Hinde Bouziane,	LIRMM, Université de Montpellier 2
Jérôme Darmont,	ERIC, Université Lumière Lyon 2
Paule-Annick Davoine	LIG, Université de Grenoble
Renaud De Landtsheer	CETIC, Charleroi, Belgique
Dimitri Durieux	CETIC, Charleroi, Belgique
Elio Goettelmann	CRP Henri Tudor, Luxembourg
Christophe Gravier	LT2C, Télécom Saint-Etienne
Anne Laurent	LIRMM, Université de Montpellier 2
Raphael Michel	CETIC, Charleroi, Belgique
Christian Sallaberry	LIUPPA, Université de Pau et des Pays de l'Adour

Comité d'organisation

Omar Boussaïd (président)	ERIC, Université Lumière Lyon 2
Előd Egyed-Zsigmond (vice-président)	LIRIS, INSA Lyon

Fadila Bentayeb	ERIC, Université Lumière Lyon 2
Danielle Boulanger	Centre Magellan, Université Jean Moulin Lyon 3
Sylvie Calabretto	LIRIS, INSA Lyon
Sylvie Cazalens	LIRIS, INSA Lyon
Samaneh Chagheri	ERIC, Université Lumière Lyon 2
Jérôme Darmont	ERIC, Université Lumière Lyon 2
Eric Disson	Centre Magellan, Université Jean Moulin Lyon 3
Cécile Favre	ERIC, Université Lumière Lyon 2
Nouria Harbi	ERIC, Université Lumière Lyon 2
Nadia Kabachi	ERIC, Université Claude Bernard Lyon 1
Sabine Loudcher	ERIC, Université Lumière Lyon 2
Pierre-Edouard Portier	LIRIS, INSA Lyon
Guilaine Talens	Centre Magellan, Université Jean Moulin Lyon 3
Caroline Wintergerst	Centre Magellan, Université Jean Moulin Lyon 3

Association Inforsid

INFORSID est une association régie par la loi de 1901 qui rassemble les chercheurs en informatique des organisations et systèmes d'information et qui a pour objectif de promouvoir les recherches effectuées dans ces domaines en faisant intervenir le plus largement possible les utilisateurs et les industriels.

INFORSID centre son activité sur un ensemble de colloques et de séminaires périodiques au cours desquels le point est fait sur l'état des recherches en matière de système d'information et une orientation est donnée pour leur prolongement.

Composition du bureau

Présidente : Régine LALEAU

Vice-présidente : Dominique RIEU

Trésorier : Philippe ROOSE

Secrétaire : Franck RAVAT

Chargé de communication : Előd EGYED-ZSIGMOND

Siège Social

INFORSID

44 chemin de la Caille

31750 Escalquens

Présidents d'honneur

Jean-Bernard CRAMPES (Toulouse)

Gilles ZURFLUH (Toulouse)

André FLORY (Lyon)

Claude CHRISMENT (Toulouse)

Michel SCHNEIDER (Clermont-Ferrrand)

Corine CAUVET (Aix-Marseille)

Chantal SOULE-DUPUY (Toulouse)

SOMMAIRE

Session - Atelier Ingénierie des exigences : modélisation, vérification et traçabilité

Session Processus : traces, fouilles et modélisation

Session Manipulation, visualisation et exploitation de modèles complexes

Session Systèmes d'information pour l'environnement

Session Ingénierie des documents et des connaissances

Session Services

Session Qualité des langages et des modèles

Conférences invitées

The Process of Process Modeling

Barbara Weber

University of Innsbruck, Austria
Barbara.Weber@uibk.ac.at

ABSTRACT. Business process models have gained significant importance due to their critical role for managing business processes. Still, process models display a wide range of quality problems. For example, literature reports on error rates between 10% and 20% in industrial process model collections. Most research in the context of quality issues in process models puts a strong emphasis on the product or outcome of the process modeling act (i.e., the resulting process models), while the process followed for creating process models is only considered to a limited extent.

The creation of process models involves the elicitation of requirements from the domain as well as the formalization of these requirements as process models. In this presentation the focus will be on the formalization of process models, which can be considered a process by itself – the process of process modeling (PPM). In particular, this presentation will shed light on the way how process models are created, present different behavioral patterns that can be observed, and discuss factors that influence the PPM, e.g., modeler-specific factors like domain knowledge or process modeling competence and task-specific factors. Moreover, it will present a specialized modeling environment, which logs all interactions of the process modeler with the modeling environment, thus, providing the infrastructure to investigate the PPM. In addition, the presentation discusses how methods like eye movement analysis, think aloud, or the analysis of bio-feedback (e.g., pulse or heart rate) might enable even deeper insights into the PPM.

La donnée est-elle soluble dans la mobiquité ?

Florence Sèdes

IRIT, Toulouse
Florence.Sedes@irit.fr

RÉSUMÉ. Face aux données de masse, liées, ouvertes, se répand la donnée personnelle qui s'expose, se partage, s'affiche, se recommande. Grâce aux avancées significatives de la capture, de l'appropriation et de la diffusion des données multimédias, de nombreux contenus disponibles incluent des caractéristiques visuelles de plus en plus riches. Les données extraites de/associées à de tels contenus peuvent être produites/attachées par des dispositifs mobiles opérés par des utilisateurs, qui peuvent les partager via des communications sans fil ou des réseaux ad hoc, et rendues accessibles via le web. Cette explosion de données multimédias ouvre de nouvelles opportunités pour le développement d'applications avancées mais également de nouveaux challenges, en termes de sécurité, de privacité, de comportements sociaux, communautaires, à l'ère du SoLoMo (So-cial, Lo-cal, Mo-bile).Face à un Internet devenu social et broadband, prospère l'écosystème endogène du smartphone : ces dispositifs intelligents tracent nos déplacements, contacts, intérêts et les trajectoires de notre vie quotidienne. Les données personnelles, massivement créées par le contrôle actif et passif des individus, ne sont pas encore exploitée par leurs propres créateurs ; ainsi émerge le besoin d'outils qui peuvent aider à rassembler, gérer et saisir la signification de toutes les données personnelles ainsi produites.

Volume, variété, véracité, vélocité des données : ainsi accessibles mais en grande majorité non exploitées, elles font émerger le besoin d'outils qui autorisent une recherche et une navigation efficaces. De multiples dispositifs doivent être pris en compte, tels que les métadonnées ou les annotations sémantiques associées aux données, la « collaboration » ou « externalisation ouverte » (crowdsourcing), les caractéristiques visuelles, les attributs spatiaux qui y sont attachés (comme la localisation GPS automatiquement attachée aux images/vidéos capturées), etc. Dans ce contexte, l'usage de certaines techniques s'avère bénéfique pour « donner du sens » à la donnée multimédia et faciliter la difficile tâche d'accès aux données pertinentes. Beaucoup de domaines d'application ont renoncé à indexer intégralement les contenus pour ne se baser que sur les métadonnées, reformulant ainsi les problématiques de data masse. Au-delà, la recherche de données multimédias pertinentes peut également tirer parti des informations additionnelles extraites du web ou d'autres sources.Les progrès dans ce domaine relèvent de différents scénarii, comme les applications exploitant des dispositifs mobiles pour capturer des événements d'intérêt en temps réel (monitoring urbain, surveillance, etc.). Les véhicules équipés de caméras dans un réseau

véhiculaire ad hoc (VANET) peuvent être amenés à échanger des informations multimédias pour informer les conducteurs des conditions de trafic, d'un danger sur la route, etc. Tout utilisateur équipé d »un dispositif de capture peut potentiellement devenir une source d'information pour une agence de presse s'il est situé dans une zone où un événement d'intérêt se déroule. De même, dans une situation d'urgence, les données multimédias capturées et transmises par différents types de dispositifs peuvent aider les services de secours à intervenir de façon plus précise, grâce à la gestion des données spatiales et temporelles obtenues à partir d'objets mobiles (e.g. croisement de trajectoires véhicules/caméras embarquées).

Actes
Sessions

Session 1

Gestion des données multimédias en environnement mobile

Recherche d'extraits vidéos par reconstitution des trajectoires de caméras mobiles à partir d'un modèle spatio-temporel - Application à la vidéosurveillance.

Dana Codreanu [1], **André Péninou** [1], **Florence Sèdes** [1]

1 IRIT, Université Paul Sabatier, 118 Route de Narbonne
Toulouse, France
dana.codreanu,andre.peninou,florence.sedes@irit.fr

RÉSUMÉ. *Le domaine d'application de ces travaux relève de l'aide aux opérateurs humains de vidéosurveillance dans l'analyse manuelle d'extraits vidéos particuliers (e.g., recherche individu, objet perdu) via la sélection automatique d'un ensemble de caméras susceptibles d'avoir filmé une scène recherchée et l'identification des séquences vidéos correspondant à chaque caméra. Cette sélection s'appuie sur un modèle de données dont la contribution consiste à rendre disponibles des données de calcul de la géométrie du champ de vue des caméras. Plutôt que de stocker cette géométrie en tant que telle et son évolution au cours du temps, celle-ci est calculée dynamiquement en fonction de caractéristiques temporelles. Nous proposons un opérateur de sélection de caméras basé sur des critères spatiotemporels de calcul des intersections entre, d'une part, un parcours donné (e.g., personne agressée), et, d'autre part, les prises de vues des caméras fixes et les trajectoires des caméras mobiles. Nous illustrons l'opérateur par des exemples.*

ABSTRACT. *The scope of application of this work concerns the assistance to the operators of videosurveillance in the search for particular videos (e.g., attack of persons, lost object) by the way of automatic identification of a set of cameras likely to have filmed a required scene. This search is based on a multi-layer modeling whose characteristic consists in rather providing calculation data of the geometry of the cameras viewpoints on the network layer than to store this geometry as such. We propose an operator of selection of cameras based on spatiotemporal criteria in order to calculate the possible intersections between, on the one hand, a given journey (attackedperson), and, on the other hand, the shootings of fixed cameras and the trajectories of the mobile cameras. We illustrate the algorithms by a use case and examples of requests.*

MOTS-CLÉS : *vidéosurveillance, trajectoire des caméras, champ de vue*

KEYWORDS: *videosurveillance, cameras trajectory, field of view*

1. Introduction

Les capteurs vidéos sont très répandus de nos jours, générant des volumes de données impressionnants. Le nombre de caméras de vidéosurveillance déployées dans les grandes villes censées assurer la sécurité des citoyens augmente régulièrement. Ainsi, le réseau de vidéosurveillance de la RATP génère plusieurs péta-octets de données de capture vidéo par jour. Les vidéos sont stockées sur des serveurs et analysées depuis des centres de contrôle. Aujourd'hui, cette analyse est purement manuelle et elle est réalisée par des opérateurs qui scrutent plusieurs écrans disposés en matrice (mur d'images).

En conséquence, de nombreux travaux sont menés pour le développement de "systèmes de vidéosurveillance intelligents". Beaucoup de travaux visent le développement d'outils d'analyse du contenu (Cucchiara, 2005) mais les problèmes de volumétrie, d'hétérogénéité de contextes d'acquisition, de qualité des enregistrements et de manque d'accessibilité à certains contenus (e.g., vidéos privées) rendent inutile voire impossible l'exécution des algorithmes d'analyse du contenu à cause des mauvaises performances ou de la faible qualité des résultats obtenus. Le besoin persiste de proposer des approches de filtrage, de modélisation, d'indexation et de recherche dans des collections de contenus vidéos sans avoir recours à une indexation exhaustive basée contenu. L'idée principale est d'utiliser des métadonnées provenant d'autres sources (e.g., données issues des capteurs, caractéristiques techniques, annotations sociales) pour générer des descripteurs ou résumés.

Avec le développement des nouvelles technologies, il est devenu facile et peu onéreux de déployer d'autres types de capteurs (e.g., capteurs GPS, boussoles, accéléromètres) associés aux caméras. Les travaux existants montrent qu'en se basant sur les métadonnées spatiales et les caractéristiques de la caméra, il est possible d'extraire des informations précieuses et précises sur la scène filmée.

Dans cet article, nous proposons une méthode qui, à partir de segments de trajectoires (géolocalisation + timestamps [1]), sélectionne les caméras susceptibles d'avoir filmé une trajectoire donnée et identifie les séquences vidéo correspondant à chaque caméra selon l'intervalle de temps. Pour cela, nous proposons de nous appuyer sur un modèle de données contenant des informations spatiotemporelles pour reconstituer les trajectoires et les géométries des champs de vue des caméras de vidéosurveillance afin de permettre la sélection des caméras fixes et mobiles. Notre approche ne nécessite pas d'accéder aux contenus des extraits vidéo ni de procéder à une indexation quelconque. Elle permet de reconstituer automatiquement des trajectoires et des géométries à partir des données stockées dans le modèle pour la trajectoire et l'intervalle de temps considérés.

L'article est structuré de la façon suivante: dans la Section 2, nous présentons une vue d'ensemble du domaine de la vidéosurveillance et nous montrons comment l'ana-

1. estampille temporelle

lyse du problème d'intersection des champs de vues des caméras avec la trajectoire de l'objet cible nous a menés à le traduire dans un problème de modélisation et interrogation de bases de données spatiotemporelles. Dans la Section 3, nous présentons un état de l'art des travaux qui utilisent l'information spatiotemporelle dans la recherche des vidéos en expliquant pourquoi ceux-ci ne peuvent pas être utilisés dans la vidéosurveillance. En se basant sur la littérature, nous proposons un modèle de données (Section 4.1) et des spécifications d'opérateurs de sélection de caméras (Section 4.2). Nous montrons quelques exemples de requêtes dans la Section 5. Nous finissons par des conclusions et des perspectives (Section 7).

2. Contexte de la vidéosurveillance

En étudiant la façon dont une requête est analysée aujourd'hui dans les systèmes de vidéosurveillance, nous avons fait plusieurs observations. Les images restituées via le mur d'images ne sont pas organisées spatialement. Très peu d'informations ou métadonnées (un identifiant de la caméra, un timestamp, éventuellement une localisation) sont disponibles pour l'opérateur qui ne peut que se référer au numéro de la caméra et à son expertise personnelle pour situer celle-ci en fonction des éléments de la requête de la victime ou de l'enquêteur dans le but de reconstituer le cheminement de la victime qui constitue le point de départ de toute recherche. Plusieurs études évoquent les problèmes de surcharge cognitive, fatigue et ennui qui peuvent entraîner des erreurs et des temps de traitement allant jusqu'à plusieurs jours (Keval, 2009).

Les enregistrements issus des systèmes de vidéosurveillance n'ont pas de sens sans informations de localisation spatiotemporelle : le besoin le plus fréquent consiste à retrouver des vidéos liées à une certaine région ou segment de rue pendant un intervalle de temps. La recherche effectuée par les opérateurs concerne donc un lieu relatif à un réseau routier ou un réseau de transport par exemple et un intervalle de temps. Une telle requête représente le point d'entrée de toute enquête. Elle décrit aussi le cheminement de l'objet cible (e.g., personne), c'est à dire un ensemble de positions à des temps donnés. La recherche porte donc sur une requête définissant une trajectoire et un intervalle de temps.

Notre but est, à partir d'une cartographie de l'ensemble des caméras de vidéosurveillance (peu importe le système auquel elles appartiennent), de pouvoir sélectionner de façon automatique les caméras qui ont pu capturer des informations pertinentes sur une trajectoire donnée.

3. État de l'art sur l'annotation et la recherche de flux vidéo

L'état de l'art sur l'analyse des vidéos basée sur les métadonnées spatiotemporelles concerne différentes approches telles que l'annotation des images, le développement de systèmes d'aide à la décision basés sur l'interrogation des informations géospatiales ou des applications pour la gestion du trafic routier. Ces approches se différentient par :

– les métadonnées sur lesquelles ces systèmes se basent : position, géométrie de la scène observée, temps, caractéristiques techniques de la caméra ;

– la prise en compte du réseau routier ou de transport ;

– la représentation de ces données continues/discrètes, e.g. champ de vue de la caméra représentée comme une région mobile (une géométrie) qui est calculée pour chaque frame [2]/minute ou seconde ;

– le(s) type(s) de requête auxquels le système permet de répondre, e.g., Key Words, Region Based, Shortest Path, Nearest Neighbor, Visibility.

Le tableau 1 présente une comparaison de quelques approches existantes en fonction des différents critères énumérés.

Dans (X. Liu *et al.*, 2009), les auteurs proposent un système (SEVA) qui annote chaque frame d'une vidéo par la localisation, le timestamp et les objets présents dans la frame. Le système est composé de: (1) une caméra vidéo, (2) une boussole numérique, (3) un système de localisation, (4) une radio wifi associée à la caméra. Les auteurs partent de l'hypothèse que tous les objets susceptibles d'être capturés sur les vidéos sont dotés d'un système qui leur permet de transmettre leur localisation (qui sera captée par la radio wifi). A partir de la localisation des caméras et de la localisation des objets, les images (frames de la vidéo) sont annotées par les objets qu'elles sont susceptibles de contenir. Des opérations d'interpolation, extrapolation, synchronisation temporelle et filtrage basées sur le champ de vue de la caméra sont réalisées pour affiner les annotations. Les auteurs partent de l'hypothèse forte que tous les objets sont munis d'un capteur qui permet au système d'avoir sa localisation à chaque moment, ce qui n'existe que dans des environnements contrôlés. Ils construisent également la géométrie du champ de vue pour chaque seconde de vidéo.

Dans (Shen *et al.*, 2011), une approche similaire à SEVA est présentée, avec les différences suivantes : (1) les objets ne doivent pas transmettre leur position et (2) leur géométrie est prise en compte et non seulement le point de localisation. Pour chaque seconde de la vidéo, les auteurs calculent le champ de vue associé à la caméra et interrogent deux bases de données extérieures (OpenStreetMaps et GeoDec) afin d'extraire les objets (e.g., bâtiments, parcs) qui se trouvent dans la scène filmée. La liste des objets est affinée en éliminant les objets qui ne sont pas visibles (en calculant une visibilité horizontale et verticale). Pour chaque objet une liste de tags est calculée à partir des ressources extérieures (e.g., localisation, mots clefs, tags extraits de la page wikipedia associée). Un classement des tags est effectué en se basant sur des critères spatiaux (la distance entre la caméra et le centre de l'objet, l'aire de l'objet dans le frame annoté, etc.). Ce système permet ensuite de retrouver des segments de vidéo à partir des requêtes textuelles en calculant une similarité entre les mots de la requête et les tags associés à chaque segment de vidéo de la base.

2. trame video

Dans (Shahabi *et al.*, 2010), les auteurs présentent un framework de visualisation des informations géospatiales liées aux caméras, images, messages (en fonction des sources de données et des contextes) pour l'aide à la décision. Leur principale contribution est d'avoir défini une architecture trois tiers qui, en se basant sur une base de donnée qui intègre des informations provenant de plusieurs sources (images satellitaires, cartes, GIS datasets, données temporelles, flux vidéos) répond à des requêtes spatiotemporelles. Un grand effort a été fait pour le développement d'une interface qui facilite une bonne visualisation et interaction (leur proposition intègre les solutions de visualisation existantes (Google Maps [3], Google Earth [4])) en améliorant l'interaction par le rajout d'une barre temporelle. Toutefois, le modèle de données et la façon dont les sources de données sont intégrées (e.g., vidéo streams) ne sont pas détaillés. Le système ne prend pas non plus en compte la géométrie des champs de vue des caméras, mais seulement les positions.

Dans (Debnath, Borcea, 2013), une approche d'annotation des images en se basant sur la localisation et l'orientation de la caméra est présentée. L'originalité est représentée par le fait qu'en plus de la localisation et des caractéristiques optiques de la caméra (angle de vue), le système proposé (TagPix) calcule une distance entre l'utilisateur et différents objets situés dans l'aire de visibilité de la caméra afin de choisir le tag le plus pertinent. Les principaux points de similarité avec notre approche sont le fait de calculer le champ de vue et la distance vues par la caméra sans avoir accès au contenu. TagPix vise l'annotation des photos donc ne considère pas la mobilité des objets et des caméras et des requêtes de type trajectoire.

Tableau 1. Comparaison des systèmes de recherche des contenus vidéo en se basant sur des métadonnées spatiotemporelles. Les abréviations de la colonne Requête du tableau correspondent aux types de requêtes présentés en début de section (KW: Key words, R: Region Based, SP: Shortest Path, NN:Nearest Neighbor, V:Visibility). Réseau R/T est une abréviation de Routier/Transport

Approche	Application	Réseau R/T	Type de Requête	Représentation de données
(X. Liu *et al.*, 2009)	Annotation	Non	KW	Geom(t, p, α, d, R)
(Shen *et al.*, 2011)	Annotation	Non	KW	Geom(t, p, α, d, R)
(Shahabi *et al.*, 2010)	Aide à la décision	O/N	R, SP, NN, V	Non
(Debnath, Borcea, 2013)	Tagging des paysages	Non	KW	Non

Le problème du développement des systèmes de videosurveillance intelligents a donné lieu à des nombreux projets de recherche comme VANAHEIM [5] qui propose

3. https://www.google.fr/maps/preview
4. http://www.google.com/earth/
5. http://www.vanaheim-project.eu/

une solution de sélection des images à montrer aux opérateurs humains en temps réel mais en se basant sur des algorithmes d'analyse du contenu vidéo. Les observations que nous pouvons faire sont que les systèmes présentés ne modélisent pas les éléments essentiels à prendre en compte dans le contexte de la vidéosurveillance : réseau routier, réseau de transport, caméras. Par ailleurs, les systèmes qui prennent en compte les géométries des scènes visualisées par les caméras construisent ces géométries pour chaque frame, ce qui est inenvisageable pour la vidéosurveillance à cause du coût de traitement et du fait que souvent le contenu vidéo n'est pas directement accessible (problèmes de droits d'accès).

En conséquence, nous allons proposer dans la suite de cet article une solution de sélection automatique des caméras fixes et mobiles dont le champ de vue a pu filmer une trajectoire donnée comme requête. Ce processus de sélection s'appuie sur un modèle de données qui permet la reconstitution des trajectoires et des géométries des champs de vue des caméras.

4. Proposition de modèle de données

Afin de réaliser notre objectif, nous proposons un modèle qui représente, d'une part les réseaux routier et de transport et d'autre part : (1) les caméras fixes et leur position sur le réseau routier, (2) les changements des caméras fixes dans le temps et (3) les caméras mobiles et leur attachement à un objet mobile donnant ainsi leur position dans le temps.

La modélisation des champs de vue des caméras par rapport à une carte nous permettra par la suite de : (a) modéliser les trajectoires des caméras mobiles, (b) modéliser les champs de vue et les distances maximales de détection des caméras fixes et mobiles et (c) sélectionner les caméras pertinentes pour une certaine trajectoire.

4.1. Représentation de données

Dans ce qui suit nous allons définir la représentation des données pour chaque composante de notre modèle.

4.1.1. Réseau Routier

Définition 1 : On définit un réseau routier G_R = (E,V) comme étant un graphe non orienté où E = $\{e_i$ / e_i=(v_j, v_k)$\}$ est un ensemble de segments de rue et V= $\{v_i\}$ est l'ensemble de croisements de ces segments (K. Liu *et al.*, 2012).

Cette modélisation nous permet de garder plusieurs niveaux de granularité du réseau routier (Sandu *et al.*, 2011). De cette façon, on peut considérer comme arête du graphe chaque segment de rue, les portions de rue entre les grandes intersections ou les rues entières. La dernière option est la plus proche de la façon dont les adresses sont exprimées dans la vie réelle par un numéro relatif à la rue entière (et non pas un segment) (e.g., Rue Alsace Lorraine no 15, Toulouse, France).

En conséquence nous allons définir deux types de positions: *une position géométrique* qui est une position 2D par rapport au système géodésique (des coordonnées GPS <lat, long>) et *une position symbolique relative au réseau routier* qui est une position similaire à l'adresse postale. Il existe des fonctions et des API publiques qui permettent de passer d'un type de position à un autre (les dénominations utilisées par Google sont Geocode et Reverse Geocode).

Les données qui composent cette couche peuvent ou non être stockées dans la même base de données que les autres. Il existe des travaux qui transforment tout le réseau routier sous forme de graphe et le stockent (e.g., (Brinkhoff, 2002)) et des travaux qui utilisent des bases de données extérieures comme OpenstreetMaps [6] (Shen *et al.*, 2011).

Définition 2 : Soit la fonction de mapping map0(position): positionsGPS $\rightarrow G_R$ qui donne la position sur le graphe G_R (e.g., Rue Montesquieu no 14) à partir d'une position GPS. (e.g., il existe plusieurs systèmes offrant ces fonctions que nous allons utiliser: geocoder [7], Google Maps [8]). Il existe bien sûr la fonction inverse $map0^{-1}$(adresse):G_R $\rightarrow$ positionsGPS (Reverse Geocoding de Google Maps).

4.1.2. Réseau de Transport

Dans le contexte de la vidéosurveillance, les requêtes sont très souvent liées au réseau de transport (e.g., sac oublié dans le bus) et il y a beaucoup de caméras mobiles installées à l'intérieur ou à l'extérieur des bus.

Définition 3 : On définit un réseau de transport G_T = (E_T ,V_T) comme étant un graphe non orienté où E_T ={ e_{ti} / e_{ti}=(v_{tj} , v_{tk})} est un ensemble de tronçons de réseau de transport et V_T= {v_{ti}} est l'ensemble des stations de bus.

Définition 4 : Soit la fonction de mapping map1(v_{ti}) : $V_T \rightarrow G_R$ qui donne la position sur le graphe G_R des noeuds du graphe G_T .

4.1.3. Objets

Les données concernant les *Objets* comprennent les positions géométriques ou symboliques des objets fixes et mobiles. Les *Objets Fixes* sont situés sur des segments de rue. Dans le cas des *Objets Mobiles*, leur position change dans le temps. Chaque objet transmet périodiquement sa position en fonction de différentes stratégies (e.g., chaque n secondes, chaque fois que l'objet change de segment) que nous ne traitons pas dans cet article. Nous supposons que les remontées sont assez fréquentes et que nous avons au moins une position par segment de rue. Dans notre modèle nous distinguons deux types d'objets mobiles : (1) objets qui se déplacent librement dans le réseau routier (e.g., voiture, personne) et (2) objets dont les trajectoires sont contraintes par une ligne (e.g., bus).

---

6. http://www.openstreetmap.org/
7. http://geocoder.us/
8. https://www.google.fr/maps/

Définition 5 : Soit MO={mo} l'ensemble d'objets mobiles. Soit id(mo)=mo_i l'identifiant de l'objet mobile. Soit la fonction TR(mo_i) = {(position(mo_i), time(mo_i))} qui extrait la trajectoire de l'objet mobile mo_i. Soit $\{position_j(mo_i)\}$ l'ensemble de positions de l'objet mobile mo_i. Soit $\{time_j(mo_i)\}$ l'ensemble des timestamps de l'objet mobile mo_i.

4.1.4. Caméras

Au-dessus de toutes ces données, nous modélisons les *Caméras*. Cette couche est composée des caméras fixes et mobiles. Les caméras fixes ont une position 2D fixée au moment de l'installation. Les caméras mobiles sont associées à un objet mobile (e.g., bus) et leur trajectoire est la même que celle de l'objet.

Les caméras de nouvelle génération possèdent des capteurs GPS incorporés et même des boussoles. Les technologies développées autour de ces caméras rendent possible l'extraction automatique des caractéristiques de prise de vue, par exemple : l'orientation, le zoom, la distance focale, etc. En se basant sur ces éléments, il est possible de modéliser le champ de vue et de tracer ces modifications dans le temps. Le champ de vue est calculé à partir de cinq paramètres principaux (Ay *et al.*, 2010) : la position, l'angle de vue, l'orientation, la distance visible et la taille du capteur.

Définition 6 : Soit l'ensemble de caméras fixes FC = {fc} / fc est une caméra fixe, id(fc)=c_i donne son identifiant, position(c_i) donne sa position et FOV(c_i) donne l'ensemble de changements de son champ de vue.

Définition 7 : Soit l'ensemble de caméras mobiles MC = {mc} / mc est une caméra mobile, id(mc)=c_i donne son identifiant, mo(c_i)=moi∈MO donne l'objet mobile auquel la caméra est associée. La trajectoire de la caméra mobile c_i sera celle de l'objet mobile avec l'identifiant mo(c_i) donc TR(c_i)= TR(mo(c_i))).

4.2. Opérateur proposé

Dans le cadre de l'aide aux opérateurs de vidéosurveillance, nous supposons la requête traduite en :

– une trajectoire spatiale constituée d'une séquence de segments tr=(u_1, u_2, ... , u_n) projetés sur le réseau routier ;

– un intervalle de temps [t_1, t_2].

Les données peuvent être extraites à partir des données d'enquête et d'outils de manipulation de données géographiques (cf. section 4.1.1). Le but est de proposer à l'agent de vidéosurveillance une liste de séquences vidéo susceptibles de contenir la trajectoire recherchée. Pour cela, il nous faut rechercher les caméras dont le champ de vue intersecte la trajectoire dans l'intervalle de temps et donc susceptibles d'avoir filmé la scène recherchée.

En conséquence, nous proposons un nouvel opérateur appelé hasSeen. L'implémentation de l'opérateur pour les caméras fixes et mobiles est différente. Nous allons

présenter par la suite séparément les deux cas de sélection des caméras fixes et des caméras mobiles.

L'opérateur hasSeen se définit ainsi. Étant donné la trajectoire spatiale composée de segments tr=(u_1,..., u_n) et l'intervalle de temps [t_1, t_2], hasSeen(tr, t_1, t_2) retourne l'ensemble des caméras c_i (1<=i<=m) associées à un segment u_k et un extrait vidéo entre deux instants t^i_{start} et t^i_{end} avec t^i_{start}∈[t_1,t_2] et t^i_{end}∈[t_1,t_2]. Chaque element de cet ensemble indique que l'extrait vidéo entre t^i_{start} et t^i_{end} de la camera c_i a filmé le segment u_k. C'est donc l'ensemble des segments vidéos des caméras susceptibles d'avoir filmé la scène recherchée.

$$hasSeen : u_1, u_2, ..., u_n, [t_1, t_2] => \begin{cases} c_1 : t^1_{start} -> t^1_{end}, u_k(1 <= k <= n) \\ c_2 : t^2_{start} -> t^2_{end}, u_k(1 <= k <= n) \\ ... \\ c_m : t^m_{start} -> t^m_{end}, u_k(1 <= k <= n) \end{cases}$$

4.2.1. Caméras Fixes

Les résultats pour les cameras fixes seront l'ensemble de triplets: R = {r = (c_i, u_k, [t_a, t_b])}, c_i ∈ FixedCameras, u_k ∈ tr, $t_1 \leq t_a < t_b \leq t_2$. L'opérateur que nous avons défini vérifie quelles sont les caméras fixes dont le champ de vue a intersecté un des segments du trajet de la requête et entre quels moments de temps (les instants t_a et t_b).

r ∈ R ≡ Il existe fov_j ∈ fov(c_i)/ (fov(c_i) est l'ensemble des instants de changement du champ de vue (Field Of View fov) de la caméra c_i) tel que :

time(fov_j) ∈ [t_1, t_2] ∧ intersects (u_k, geometry(position(c_i), fov_j)) ∧ t_a = time(fov_j) ∧ t_b = min (time (succ(fov_j)), t_2)	*dans le cas où le point de changement est à l'intérieur de l'intervalle [t_1, t_2], si la géométrie du champ de vue correspondant intersecte l'un des segments du trajet de la requête, alors l'intervalle de temps commence au moment du changement du champ de vue et se finit au prochain changement ou à la fin de l'intervalle de la requête*
∨	
time(fov_j)<t_1 ∧ t_1<=time(succ(fov_j)) ∧ intersects (u_k, geometry(position(c_i), fov_i)) ∧ t_a = t_1 ∧ t_b = min (time (succ(fov_i)), t_2)	*dans le cas où le point de changement est avant t_1, et son suivant est après t_1, si la géométrie du champ de vue correspondant intersecte l'un des segments du trajet de la requête, alors l'intervalle de temps commence au moment t_1 et se finit au prochain changement ou à la fin de l'intervalle de la requête*

4.2.2. Caméras Mobiles

L'opérateur que nous avons défini retrouve quelles sont les caméras associées à des objets mobiles dont une position connue intersecte un des segments du trajet de la requête et entre quels moments de temps il peut l'avoir intersecté (instants t_a et t_b). Dans ce cas, la mobilité des caméras ne permettent pas de calculer l'intersection précise entre le champ de vue de la caméra et les segments de la requête (par exemple une caméra placée sur le côté extérieur droit d'un bus). Le résultat obtenu indique uniquement que la caméra mobile se trouvait sur un segment de la requête dans l'intervalle de temps de la requête.

Les résultats pour les caméras mobiles seront l'ensemble de triplets: R = {r = (c_i, u_k, [t_a, t_b])}, $c_i \in$ MC, $u_k \in$ tr, $t_1 \leq t_a < t_b \leq t_2$.

r $\in$ R $\equiv$ Il existe $mp_j \in$ TR(mo(c_i)) (il existe une position dans la trajectoire de l'objet mobile auquel la caméra mobile c_i est attachée) tel que :

[time($mp_j(mo_i)$) $\in$ [t_1,t_2]	
$\wedge$ intersects (position($mp_j(mo_i)$), u_k)	
$\wedge$ (not intersects(	
prec(position($mp_j(mo_i)$)), tr)	
$\vee$ (intersects(	
prec(position($mp_j(mo_i)$)), tr)	
$\wedge$ prec(time($mp_j(mo_i)$))<t_1))	
$\wedge$ t_a=max (prec(time($mp_j(mo_i)$)), t_1)	*dans le cas où la remontée de la position de l'objet est sur le trajet de la requête et est à l'intérieur de l'intervalle [t_1, t_2] et la position d'avant n'est pas sur le trajet de la requête, alors l'intervalle de temps commence au maximum entre le moment de la dernière remontée et t_1 et se finit à la prochaine remontée ou à la fin de l'intervalle de la requête*
$\wedge$ t_b=min(succ(time($mp_j(mo_i)$)),t_2)]	
$\vee$ **// deuxième partie de l'opérateur**	
[t_1<= time($mp_j(mo_i)$)	
$\wedge$ time($mp_j(mo_i)$)<= t_2	
$\wedge$ intersects (position($mp_j(mo_i)$), u_k)	
$\wedge$ intersects(	
(prec(position($mp_j(mo_i)$))), tr)	
$\wedge$ t_a =time($mp_j(mo_i)$)	
$\wedge$ t_b=min(succ(time($mp_j(mo_i)$)), t_2)]	*dans le cas où la remontée de la position de l'objet est sur le trajet de la requête et est à l'intérieur de l'intervalle [t_1, t_2] et la position d'avant est aussi sur le trajet de la requête, alors l'intervalle de temps commence au moment de la remontée et se finit à la prochaine remontée ou à la fin de l'intervalle de la requête*

Nous définissons les fonctions et prédicats :

– *geometry(point p, fov f):* region calcule la géométrie du champ de vue d'une caméra à partir de sa position et de ses caractéristiques optiques (Ay *et al.*, 2010).

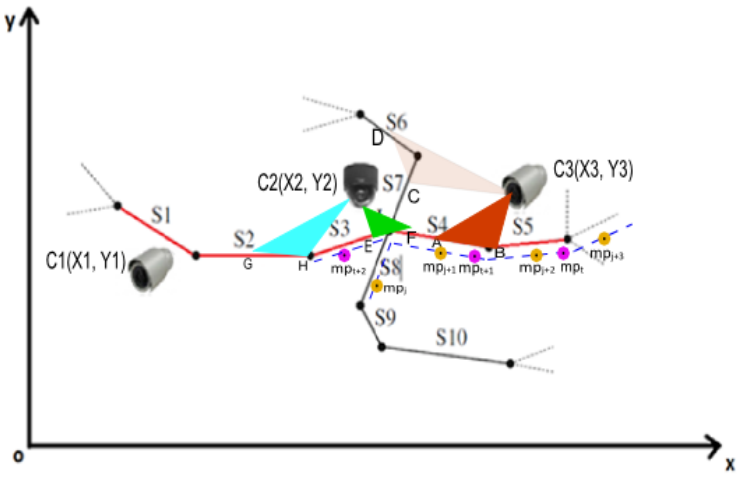


Figure 1. Schéma d'un réseau routier filmé par trois caméras fixes

Plutôt que de stocker cette géométrie en tant que telle et de la générer pour chaque frame de la vidéo comme le font les systèmes présentés dans l'état de l'art, elle est calculée en dynamique en fonction de caractéristiques temporelles au moment de la requête.

– *intersects(*line *seg,* region *g): boolean* permet de vérifier si la géométrie g intersecte un segment de rue.

– *intersects(*point *p,* line *seg): boolean* permet de vérifier si un point p intersecte un segment de rue.

– *intersects(*point *p, set(*line*) tr): boolean* permet de vérifier si un point p intersecte un ensemble de segments ; elle retourne vrai si $\exists\ seg_i \in$ tr / intersects (p, seg_i).

5. Exemples de requêtes

5.1. Caméras fixes

Supposons le schéma de la Figure 1 illustrant les localisations des caméras fixes C1, C2 et C3. Le réseau routier est représenté par les deux rues (S1-S5 et S6-S10). Supposons le trajet de la requête TR= S1,..,S5 et l'intervalle de la requête I= [t_1, t_2].

La Figure 2 présente les prises de vue des caméras C_2 et C_3 en fonction du temps. Les différents moments où les champs de vue des caméras C_2 et C_3 changent (voir Figure 1) sont marqués en couleurs correspondant aux géométries de la Figure 1 (e.g., au moment $time_j(fov(c_3))$ le champ de vue devient ABC_3).

Supposons l'intervalle de la requête [t_1, t_2] avec $time_j(fov(C_3))$<t_1<$time_k$ (fov(C_2)) et $time_{j+1}(fov(C_3))$< t_2.

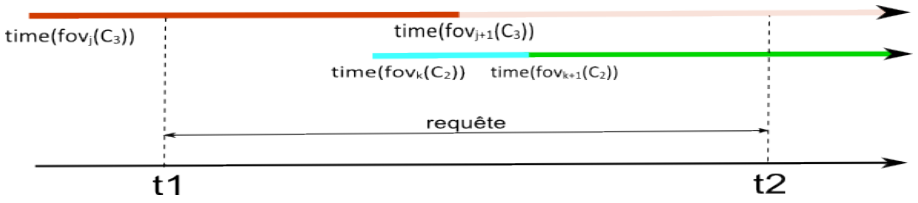


Figure 2. Les moments de changement de champ de vue et l'intervalle de la requête

La sélection des caméras fixes est réalisée par l'algorithme 1. Nous allons expliquer cet algorithme par l'intermédiaire de l'exemple illustré dans la Figure 1. Les premières lignes de l'algorithme (1-3) représentent une étape de filtrage. De toutes les caméras de la base de donnée nous allons sélectionner celles qui se situent à une distance maximale inférieure ou égale à la distance visible maximale des caméras de la base. Dans notre cas les seules caméras qui ont possiblement filmé les segments de la requête sont C_1, C_2 et C_3.

Pour chacune des caméras sélectionnées lors de la première étape, nous allons chercher les moments où celles-ci ont modifié leur champ de vue (field of view fov) (lignes 4,5 de l'algorithme). Les lignes 6-19 traitent les deux cas possibles: le changement est entre t_1 et t_2 (e.g., time($fov_k(C_2)$) ou le changement est avant t_1 (e.g., time($fov_j(C_3)$)). Les géométries des champs de vues sont construites et leurs intersections avec les segments de la requête sont evaluées.

Le résultat de la requête pour notre exemple est:
{(C_2, S_2, [time($fov_k(C_2)$), time($fov_{k+1}(C_2)$)]), (C_2, S_3, [time($fov_{k+1}(C_2)$), t_2])), (C_2, S_4, [time($fov_{k+1}(C_2)$), t_2))), (C_3, S_4, [t_1, time($fov_{j+1}(C_3)$)])}.

5.2. Caméras mobiles

Considérons le même schéma de la figure 1. Le but est maintenant de sélectionner les caméras mobiles susceptibles d'avoir filmé la trajectoire de la requête TR=S1, S2, S3, S4, S5. Cette sélection est faite selon l'algorithme 2.

Supposons deux objets mobiles ayant les trajectoires marquées en pointillé sur la figure (S8, S4, S5, ...). Nous supposons que l'objet envoie au moins une remontée mp_j (mobile position) contenant sa position et un timestamp par segment de rue. En considérant chaque segment de la rue et chaque objet mobile (lignes 1-2 de l'algorithme), la fonction $filtrer(mo_i, u_k, [t_1, t_2])$ va tester les cas expliqués dans 4.2.2 : la position de l'objet est sur la trajectoire de la requête et entre t_1 et t_2 (mp_t, mp_t, mp_{j+1}, mp_{j+2} comme illustré dans la Figure 3) et la position d'avant intersecte aussi (mp_{j+1} et mp_{j+2}) ou la position d'avant n'intersecte pas la trajectoire (mp_j et mp_{J+1}) ou elle intersecte mais avant t_1 (mp_t et mp_{t+1}).

Le résultat est {(obj_i, S_4, [t_1,time(mp_{j+1})]), (obj_i, S_5, [time(mp_{j+1}), t_2)], (obj_{i+1}, S_4, [time(mp_t),t_2)])}

Algorithm 1: L'algorithme de sélection des caméras fixes qui ont intersecté la trajectoire de la requête

Entrées: Une suite de segments de rue: u_k et un intervalle de temps: $[t_1, t_2]$.
Sorties: La liste des caméras fixes qui ont vu la trajectoire donnée

pour *chaque* u_k *de la requête* **faire**
  $listeCam \leftarrow extraireCamDist(u_k, max(FOV.visibleDistance))$
fin
pour *chaque* c_i *de listeCam* **faire**
  pour *chaque* $(fov_j(c_i))$ **faire**
    si *time*$(fov_j(c_i)) >= t_1$ *et time*$(fov_j(c_i)) <= t_2$ **alors**
      $geometry_{ij} \leftarrow construire\_polygone(fov_j(c_i))$;
      pour *chaque* u_k *de la requête* **faire**
        si $geometry_{ij}$ *intersecte* u_k **alors**
          $ajouter(c_i, u_k, [time(fov_j), min(succ(time(fov_j)), t_2)])$;
        fin
      fin
    fin
    si *time*$(fov_j(c_i)) < t_1$ *et* $t_1 <=$ *time*$(succ(fov_j(c_i)))$ **alors**
      $geometry\_ij \leftarrow construire\_polygone(fov_j(c_i))$;
      pour *chaque* u_k *de la requête* **faire**
        si $geometry_{ij}$ *intersecte* u_k **alors**
          $ajouter(c_i, u_k, [t_1, min(time(succ(fov_j)), t_2)])$;
        fin
      fin
    fin
  fin
fin

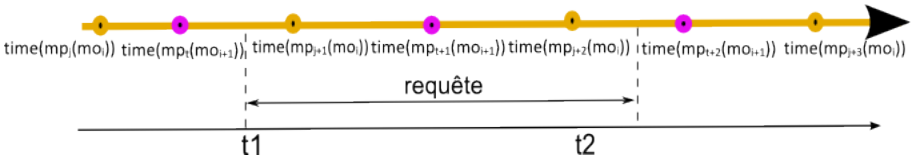


Figure 3. Les moments des trajectoires des objets mobiles et l'intervalle de la requête

6. Implémentation et validation

La figure 4 illustre l'architecture générique que nous avons implémentée et qui vise à faciliter la conception des outils forensic [9]. Nous allons brièvement présenter le

9. http://en.wikipedia.org/wiki/Forensic_science

Algorithm 2: L'algorithme de sélection des caméras mobiles qui ont intersecté la trajectoire de la requête

Entrées: Une suite de segments de rue: $u_k$et un intervalle de temps: $[t_1, t_2]$.
Sorties: La liste des caméras mobiles qui ont vu la trajectoire donnée

pour *chaque* u_k **faire**
  pour *chaque* mo_i **faire**
    $listeObjMobiles \leftarrow ajouter(filtrer(mo_i, u_k, [t_1, t_2]))$;
  fin
fin
pour *chaque* mo_i*.id de listeObjMobiles* **faire**
  $listeCameras \leftarrow selectionnerCameras(mo_i.id)$;
fin

choix d'implémentation que nous avons fait (dans la plupart des cas imposé par les contraintes du projet) mais qui n'est pas certainement le seul possible :

– **Terminal Interface (TI)** : Permet de saisir des requêtes à partir d'une interface basée sur l'API Google Maps.

– **Query Interpreter (QI)** : Transforme la requête de l'utilisateur dans une requête spatiotemporelle comme celle décrite dans la section 4.2 (une séquence de segments et un intervalle de temps).

– **SQL Query Generator (SQLG)** : Ce module implémente les deux algorithmes décrits dans la section antérieure. Il génère et envoie à la base de données les requêtes nécessaires pour le calcul des géométries des champs de vues des caméras fixes et leur intersection avec les segments de la requête.

– **Spatiotemporal database(DB)** : Ce module est responsable du stockage des données à partir desquelles le module SQLG reconstitue les trajectoire des objets mobiles et les géométries des champs de vue des caméras. Nous avons utilisé Oracle 11g (avec l'extension Spatial) pour implémenter ce module.

Les modules QI et SQLG sont implémentés en Java et le TI est implémenté en HTML/Javascript. Nous avons utilisé JSP et Ajax pour connecter l'interface avec les deux autres modules. Les tests sur des données expérimentales et données réelles sont en cours de mise en oeuvre.

7. Conclusion et perspectives

Dans cet article nous avons proposé une approche de sélection automatique des caméras susceptibles d'avoir filmé une trajectoire. Nous avons pu valider la contribution applicative de notre travail qui est de montrer la faisabilité et les bénéfices de l'utilisation des métadonnées (e.g., de géolocalisation, caractéristiques optiques de la caméra, réseau routier, réseau de transport) associées aux dispositifs de prise de vue dans un modèle qui constitue la base d'un système d'assistance aux opérateurs de vi-

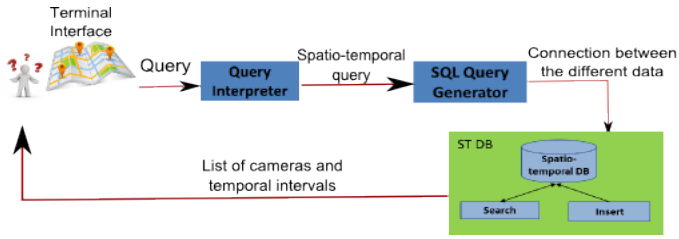


Figure 4. Architecture de l'outil proposé

déo surveillance. Cette approche permet de s'affranchir de l'indexation et de l'accès aux contenus vidéo, dans une première phase de recherche qui opère un filtrage important. Le besoin opérationnel exprimé dans cet article a été extrait de l'analyse du besoin et des entretiens avec les opérationnels (e.g., RATP, SNCF) réalisés dans le cadre du projet ANR METHODEO [10]. La solution proposée a été également implémentée et validée dans le cadre du projet dans un contexte industriel.

Les opérateurs de sélection proposés se basent sur un modèle de données spatiales et temporelles des caméras de vidéosurveillance dans le cadre d'un réseau de transport en commun urbain qui permet la reconstitution des trajectoires des objets mobiles des caméras mobiles (associées à des objets mobiles) et des positions des caméras fixes ; chaque caméra se voit associé un champ de vue dont la géométrie est calculée au moment de la requête pour les caméras fixes susceptibles d'avoir capturé des vidéos pertinentes pour la trajectoire cible. L'approche proposée, renforcée par des solutions d'indexation et de stockage optimisées et complétée par des solutions d'analyse du contenu vidéo, peut constituer la base d'outils de Forensic (Sèdes *et al.*, 2012) si recherchés dans le cadre de la vidéosurveillance.

Le résultat de l'opérateur proposé est une liste de segments vidéo dont les intervalles de temps peuvent se recouvrir. Une perspective concerne donc l'ordonnancement des résultats en fonction de la distance des caméras par rapport aux segments de la requête par exemple.

Pour l'instant notre modèle considère seulement les caméras situées en environnement outdoor qui ont des positions GPS qui permettent de faire la projection sur le graphe du réseau routier. Des travaux comme ceux de (Jensen *et al.*, 2009) modélisent également les environnements indoor sous forme de graphe (les noeuds du graphe sont représentés par des pièces et les segments par des connexions entre les pièces e.g., portes). Une perspective intéressante de notre travail est d'étendre le modèle pour prendre en compte les réseau de caméras à l'intérieur de stations de métro

10. http://www.agence-nationale-recherche.fr/suivi-bilan/ingenierie-procedes-securite/concepts-systemes-et-outils-pour-la-securite-globale/detail-des-projets-finances-csosg/?tx_lwmsuivibilan_pi2[CODE]=ANR-10-SECU-0006

ou des gares par exemple, le principal verrou étant de définir les positions absolues et les positions relatives des caméras.

Une autre perspective consiste à utiliser de manière plus poussée la modélisation des réseaux urbains proposée pour améliorer la recherche, par exemple si des événements se passent sur des lignes de réseaux de transport.

Bibliographie

Ay S. A., Kim S. H., Zimmermann R. (2010). Generating synthetic meta-data for georeferenced video management. In *Proceedings of the 18th sigspatial international conference on advances in geographic information systems*, p. 280–289. New York, NY, USA, ACM. Consulté sur http://doi.acm.org/10.1145/1869790.1869830

Brinkhoff T. (2002, jun). A framework for generating network-based moving objects. *Geoinformatica*, vol. 6, n° 2, p. 153–180. Consulté sur http://dx.doi.org/10.1023/A:1015231126594

Cucchiara R. (2005). Multimedia surveillance systems. In *Proceedings of the third acm international workshop on video surveillance and sensor networks*, p. 3–10. New York, NY, USA, ACM.

Debnath H., Borcea C. (2013). Tagpix: Automatic real-time landscape photo tagging for smartphones. In *6th international conference on mobile wireless middleware, operating systems, and applications.*

Jensen C. S., Lu H., Yang B. (2009). Graph model based indoor tracking. In *Tenth international conference on mobile data management: Systems, services and middleware, 2009*, p. 122-131.

Keval H. U. (2009). *Effective design, configuration, and use of digital cctv*. Thèse de doctorat non publiée, University College London.

Liu K., Li Y., He F., Xu J., Ding Z. (2012). Effective map-matching on the most simplified road network. *In Proceedings of the 20th International Conference on Advances in Geographic Information Systems*, p. 609-612.

Liu X., Corner M., Shenoy P. (2009). Seva: Sensor-enhanced video annotation. *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 5, n° 3, p. 1–26.

Sandu P. I., Zeitouni K., Oria V., Barth D., Vial S. (2011). Indexing in-network trajectory flows. *The VLDB Journal*, vol. 20, n° 5, p. 643-669.

Sèdes F., Sulzer J., Marraud D., Mulat C., Cepas B. (2012). Intelligent video surveillance systems. In J.-Y. Dufour (Ed.),, chap. A Posteriori Analysis for Investigative Purposes. Wiley.

Shahabi C., Banaei-Kashani F., Khoshgozaran A., Nocera L., Xing S. (2010). Geodec: A framework to visualize and query geospatial data for decision-making. *IEEE Multimedia*, vol. 17, n° 3, p. 14-23.

Shen Z., Arslan Ay S., Kim S. H., Zimmermann R. (2011). Automatic tag generation and ranking for sensor-rich outdoor videos. In *Proceedings of the 19th acm international conference on multimedia*, p. 93–102. New York, NY, USA, ACM.

Détection de flux de contrôle illégaux dans les Smartphones

Mariem Graa[1,2] — Nora Cuppens-Boulahia[1] — Frédéric Cuppens[1] — Ana Cavalli[2]

[1] Telecom-Bretagne
2 Rue de la Chataigneraie,35576 Cesson Sevigne - France
{mariem.benabdallah,nora.cuppens,frederic.cuppens}@telecom-bretagne.eu

[2] Telecom-SudParis
9 Rue Charles Fourier, 91000 Evry - France
{mariem.graa, ana.cavalli}@it-sudparis.eu

RÉSUMÉ. *La sécurité dans les systèmes embarqués tels que les smartphones exige une protection des données privées manipulées par les applications tierces. Certains mécanismes utilisent des techniques d'analyse dynamique basées sur le « data-tainting » pour suivre les flux d'informations dans le programme. Mais ces techniques ne peuvent pas détecter les flux de contrôles qui utilisent des instructions conditionnelles pour transférer implicitement les informations. En particulier, les applications malveillantes peuvent contourner le système Android et obtenir des informations sensibles à travers les flux de contrôles. Nous proposons une amélioration de l'analyse dynamique qui propage la teinte tout au long des dépendances de contrôles en utilisant les données fournies par l'analyse statique dans les systèmes Android. Notre approche réussit à détecter des attaques de contrôle de flux sur les smartphones.*

ABSTRACT. *Security in embedded systems such as smartphones requires protection of private data manipulated by third-party applications. Many mechanisms use dynamic taint analysis techniques for tracking information flow in software. But these techniques cannot detect control flows that use conditionals to implicitly transfer information from objects to other objects. In particular, malicious applications can bypass Android system and get privacy sensitive information through control flows. We propose an enhancement of dynamic taint analysis that propagates taint along control dependencies by using the static analysis in embedded system such as Google Android operating system. Our approach allows the detection of control flow attacks on smartphones.*

MOTS-CLÉS : *Smartphones, Contrôle de flux, Tainting, Analyse statique, Analyse dynamique*

KEYWORDS: *Smartphones, Control flow, tainting, static analysis, dynamic analysis*

1. Introduction

Ces dernières années ont connu une augmentation de l'utilisation des systèmes embarqués tels que les Smartphones. D'après un récent rapport de Gartner (Egham, November 2010), 417 millions de téléphones mobiles ont été vendus au troisième trimestre de 2010, ce qui correspond à 35% d'augmentation par rapport à 2009. Les utilisateurs de Smartphones téléchargent des applications tierces pour rendre leurs téléphones plus utiles. Android Market annonce que le taux de téléchargement des applications tierces a dépassé 10 milliards d'applications en Décembre 2011 (Haselton, Decembre 2011). Ces applications peuvent être utilisées pour accéder et manipuler les données privées stockées dans les Smartphones. Selon une étude réalisée par Lookout Mobile Security, le nombre de malwares est en forte progression sur les plates-formes mobiles (Laporte, August 2011). Lookout Mobile Security prend l'exemple d'Android qui comptait 80 applications contenant du code malveillant en Janvier 2011. Ce chiffre a été multiplié par cinq en Juin 2011. Lookout Mobile Security estime que près de 500 000 personnes ont été victimes d'un malware sur Android au premier semestre de 2011. Dans l'étude présentée à la Conférence Black Hat, Daswani (Wilson, July 2011) a analysé le comportement de 10 000 applications Android et a montré que plus de 800 provoquent la fuite des données privées à un serveur non autorisé. Par conséquent, il est nécessaire de prévoir des mécanismes de sécurité adéquats pour contrôler la manipulation des données privées par des applications tierces. Plusieurs mécanismes sont utilisés pour protéger les données privées dans un système Android, telle que l'analyse dynamique qui est implémentée dans TaintDroid (Enck *et al.*, 2010). Le principe de l'analyse dynamique est d'associer une teinte aux données privées dans un système puis de propager cette teinte aux autres données qui en dépendent lors de l'exécution du programme pour suivre le flux d'information. Il existe deux types de flux d'information : les flux explicites et les flux implicites (flux de contrôle). Un exemple de flux explicite se produit lors d'une affectation $x = y$, où on observe un transfert explicite d'une valeur de x à y. Un exemple de flux de contrôle est illustré dans la Figure 1, où il n'y a pas de transfert direct de la valeur de a à b, mais lorsque le code est exécuté, b obtient la valeur de a.

```
1.boolean b = false;
2.boolean c = false;
3.if (!a)
4. c = true;
5.if (!c)
6.  b = true;
```

Figure 1. *Exemple de flux implicite.*

TaintDroid ne propage pas la teinte à travers les flux de contrôles ce qui provoque un problème d'under tainting : le processus de teintage tel que défini par Taintroid engendre des faux négatifs. Les applications malveillantes peuvent contourner un sys-

tème Android et obtenir des données privées en exploitant les flux de contrôles. Dans cet article, nous proposons une approche exhaustive et opérationnelle qui propage la teinte tout au long des flux de contrôles. Elle combine l'analyse statique et dynamique pour résoudre le problème d'under tainting. Notre approche détecte les attaques qui provoquent la fuite des données privées en exploitant les flux de contrôles au cours de l'exécution des applications Android. Nous allons, dans un premier temps, présenter dans la Section 2 des attaques exploitant des dépendances de contrôles que TaintDroid ne peut pas détecter. Dans la Section 3, nous analysons les travaux sur l'analyse statique et dynamique. Nous décrivons dans la Section 4 l'approche de TaintDroid. Nous présentons notre solution basée sur une approche hybride qui améliore la fonctionnalité de TaintDroid pour tracer les flux de contrôles dans la Section 5. Dans la section 6, nous montrons que notre approche détecte les attaques présentées dans la Section 2. Dans la section 7, nous présentons les résultats de test et d'évaluation de notre approche. La Section 8 est consacrée à une discussion portant sur les « faux positifs ». La Section 9 conclut l'article.

2. Attaques de contrôle de flux

Sarwar *et al.*(Sarwar *et al.*, 2013) présentent des attaques de contrôle de flux. Ces attaques visent le mécanisme de teintage dans TaintDroid. Le but de l'attaquant qui est le fournisseur de l'application malveillante est d'obfusquer le code et de tromper le processus de teintage dans TaintDroid pour qu'il ne teinte pas des données privées. Il joue sur la structuration du code (des flux) puisque TaintDroid ne propage pas la teinte dans les flux de contrôles. Sarwar *et al.* montrent expérimentalement le taux de réussite de ces attaques pour contourner la propagation de la teinte dans TaintDroid.

Algorithm 1 Attaque d'encodage

$X \leftarrow Private\_Data$
for each $x \in X$ **do**
  for each $s \in AsciiTable$ **do**
    if $(s == x)$ **then**
      $Y \leftarrow Y + s$
    end if
  end for
end for
$Send\_Network\_Data(Y)$

L'Algorithme 1 présente la première attaque. La variable X contient la donnée privée. L'attaquant utilise une boucle for pour parcourir les caractères de X et les comparer aux symboles de table ASCII. Les bons caractères sont concaténés dans la chaine de caractère Y. A la fin de cette boucle l'attaquant réussit à savoir la valeur de la donnée privée stockée dans Y. Comme TaintDroid ne propage pas la teinte dans les flux de contrôles, la variable Y n'est pas teintée. Elle est envoyée à travers le réseau

sans être détectée.

Algorithm 2 Attaque de compteur

$X \leftarrow Private\_Data$
for each $x \in X$ **do**
  $n \leftarrow CharToInt(x)$
  $y \leftarrow 0$
  for $i = 0$ to n **do**
    $y \leftarrow y + 1$
  end for
  $Y \leftarrow Y + IntToChar(y)$
end for
$Send\_Network\_Data(Y)$

L'Algorithme 2 présente la deuxième attaque. La donnée privée est affectée à la variable X. L'attaquant utilise de l'encodage de code pour convertir chaque caractère de X en un entier. En utilisant un compteur, il réussit à savoir la valeur de cet entier. Ensuite, il reprend la technique d'encodage pour convertir cet entier en un caractère. Les caractères obtenus sont concaténés dans Y. A la fin de l'exécution de cet algorithme la variable Y contient la donnée privée mais elle n'est pas teintée car TaintDroid ne détecte pas les flux de contrôles. Elle est envoyée à travers le réseau sans être détectée.

Algorithm 3 Attaque d'exception

$X \leftarrow Private\_Data$
for each $x \in X$ **do**
  $n \leftarrow CharToInt(x)$
  $y \leftarrow 0$
  while $y < n$ **do**
    $Try\{$
    $Throw\_New\_Exception()\}$
    $Catch(Exception\ e)\{$
    $y \leftarrow y + 1\}$
  end while
  $Y \leftarrow Y + IntToChar(y)$
end for
$Send\_Network\_Data(Y)$

L'Algorithme 3 présente une attaque basée sur les exceptions. La variable n contient un entier qui correspond à la conversion d'un caractère de la donnée privée. L'attaquant lance une exception n fois.
Il traite l'exception dans le bloc "catch" en incrémentant la variable y pour atteindre la valeur de n. La conversion de y en caractère permet de savoir un caractère de la donnée privée. En concaténant tous les caractères trouvés, l'attaquant réussit à connaitre la valeur de la donnée privée. TaintDroid ne propage pas la teinte dans les exceptions

utilisées dans les flux de contrôles. Par conséquent, l'envoi de la variable Y contenant la donnée privée n'est pas détecté par TaintDroid. Nous présentons dans la section suivante les approches existantes utilisant l'analyse statique et dynamique qui peuvent être utilisées pour détecter des attaques définies par les dépendances de contrôles.

3. Travaux de recherche associés

Data Tainting permet de tracer la propagation des données dans un système. Le principe de ce mécanisme est de teinter (associer un tag) les données dans un programme et de propager la teinte aux objets dépendants. Il est utilisé pour la détection de vulnérabilités, la protection des données sensibles et plus récemment, pour l'analyse de malware. Une vulnérabilité est détectée lorsqu'une donnée teintée est utilisée dans un « Taint Sink ». Le data tainting est implémenté dans les interpréteurs (Wall *et al.*, 2000),(Hunt et Thomas, 2000) pour suivre les données sensibles. Il est utilisé pour analyser dynamiquement le code binaire (Newsome et Song, 2005),(Cheng *et al.*, 2006),(Qin *et al.*, 2006),(Yin *et al.*, 2007) en l'instrumentant pour suivre la propagation de la teinte. Ainsi, ce mécanisme dégrade les performances (temps d'exécution) du système ce qui ne favorise pas leur utilisation dans des applications exécutées en temps réel. Plusieurs approches étudient la protection des données personnelles sur les smartphones. Les approches basées sur le contrôle d'accès (Enck *et al.*, 2009 ; Nauman *et al.*, 2010 ; Conti *et al.*, 2011 ; Bugiel *et al.*, 2011 ; Ongtang *et al.*, 2010) assurent que seules les applications qui ont les droits nécessaires peuvent accéder aux données privées. Mais, ces approches n'assurent pas la protection des données de bout en bout. Elles sont complétées par les approches qui contrôlent le flux des données. TaintDroid (Enck *et al.*, 2010) implémente l'analyse dynamique pour suivre les flux de données dans les applications en temps réel. Enck *et al.* s'inspirent des travaux précédents, mais ils adressent des défis différents spécifiques aux smartphones comme les ressources limitées. AppFence (Hornyack *et al.*, 2011) est une extension de Taintdroid qui détecte et bloque l'envoi des données privées en dehors du système. Ces approches permettent de suivre seulement les flux explicites et ils ne détectent pas les flux de contrôles. Ainsi, ils ne peuvent pas détecter les attaques liées aux dépendances de contrôles qui visent la vie privée des utilisateurs. Cavallaro *et all.* (Cavallaro *et al.*, 2008) décrivent les techniques d'évasion contre l'analyse dynamique des flux d'informations. Ces attaques d'évasion utilisent les dépendances de contrôles et provoquent la fuite des données privées. Cavallaro *et al.* pensent qu'il est nécessaire de raisonner sur les affectations des variables dans les structures conditionnelles. Nous avons suivi ce raisonnement dans l'implémentation des règles qui définissent la propagation de la teinte. Certaines approches existent dans la littérature pour suivre les flux de contrôles (Egele *et al.*, 2007), (Song *et al.*, 2008), (Kang *et al.*, 2011), (Nair *et al.*, 2008). Elles combinent l'analyse statique et dynamique pour identifier correctement les flux implicites et pour détecter la fuite des données sensibles. DTA ++ (Kang *et al.*, 2011) étend l'analyse dynamique pour suivre les flux de contrôles. Cependant, DTA ++ est évaluée uniquement sur les applications bénignes et il n'est pas testé sur les programmes malveillants. En outre, ces approches ne sont

pas implémentées dans les systèmes embarqués tel que les smartphones. L'analyse statique implémentée dans les smartphones (Egele *et al.*, 2011 ; Chin *et al.*, 2011 ; Fuchs *et al.*, 2009) permet de détecter les fuites de données, mais elle ne peut pas capturer toutes les configurations durant la phase d'exécution. Nous nous sommes inspirés de ces travaux, mais nous avons utilisé une approche qui combine l'analyse statique et l'analyse dynamique pour détecter les attaques exploitant les dépendances de contrôles dans le système Android. Fenton (Fenton, 1974) a proposé une Machine "Data Mark", un modèle abstrait, pour gérer les flux de contrôles. Il donne une description formelle de son modèle et une preuve de correction en termes de flux d'information. Aries (Brown et Knight Jr, 2001) interdit l'écriture à un emplacement particulier dans la branche conditionnelle lorsque la classe de sécurité associée à cet emplacement est égale ou moins restrictive que celles de compteur de programme. L'approche d'Aries utilise uniquement les classes de sécurité de niveau haut et bas. Denning (Denning, 1975) améliore le mécanisme défini par Fenton en terme de temps d'exécution avec un mécanisme se déclenchant pendant la compilation pour détecter tous les flux de contrôles. Il insère des instructions supplémentaires si la branche est prise ou pas pour déterminer la classe de sécurité modifiée afin de refléter le flux d'informations. Nous nous inspirons de l'approche de Denning, mais nous avons formellement défini un ensemble de règles de propagation de la teinte afin d'éviter les attaques exploitant les dépendances de contrôles. Nous décrivons dans la Section suivante plus en détail l'approche de TaintDroid.

4. Système de contrôle de flux : Taintroid

Les applications tierces installées sur un smartphone peuvent extraire des données privées de l'utilisateur. TaintDroid est une extension de la plateforme Android. Il est implémenté dans la machine virtuelle de smartphone.

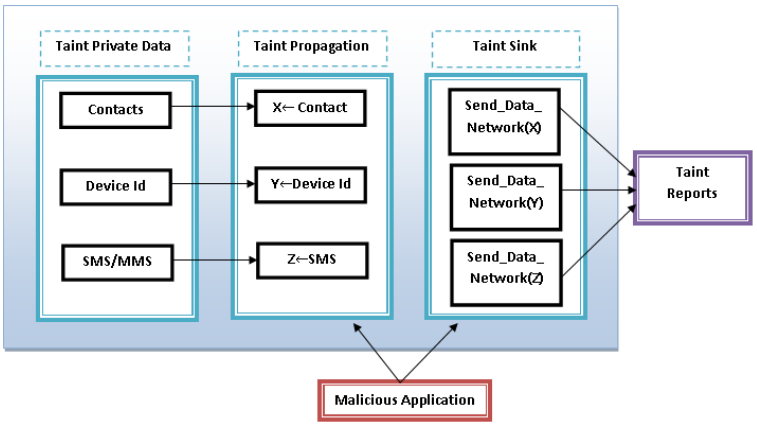


Figure 2. *Processus de TaintDroid*

TaintDroid utilise le mécanisme de data tainting et l'analyse dynamique pour suivre les flux explicites en temps réel et pour contrôler la manipulation des données privées par les applications tierces. Le processus de TaintDroid est présenté dans la Figure 2. D'abord, il associe une teinte aux données privées. Ensuite, il suit la propagation des données teintées. Enfin, il détecte une vulnérabilité si une donnée teintée est utilisée dans un emplacement sensible (taint sink) qui permet d'envoyer la donnée en dehors du système. L'inconvénient de TaintDroid est qu'il ne détecte pas les flux de contrôles. Donc il ne peut pas détecter les attaques exploitant les dépendances de contrôles. Nous décrivons notre approche plus en détail dans la section suivante.

5. Détection des flux de contrôles dans les Smartphones

Comme précisé dans la Section 4 TaintDroid ne propage pas la teinte à travers les flux de contrôles ce qui cause un problème d'under tainting. Dans cette section, nous commençons par donner une spécification formelle du problème d'under tainting. Ensuite, nous présentons notre solution formelle basée sur deux règles qui définissent la politique de teintage. Enfin, nous présentons notre solution technique qui propage la teinte tout au long des dépendances de contrôles en utilisant les deux règles de propagation pour résoudre le problème d'under tainting.

5.1. *Spécification formelle du problème d'under tainting*

Denning (Denning, 1976) définit un modèle de flux d'informations comme suit :

$$FM =< N, P, SC, \oplus, \rightarrow > .$$

– N est un ensemble d'objets de stockage logique (fichiers, variables,...).

– P est un ensemble de processus qui sont exécutés par les agents responsables de tous les flux d'informations.

– SC est un ensemble de classes de sécurité qui sont affectées aux objets de N. SC est un ensemble fini qui a une limite inférieure L associée aux objets de N.

– l'opérateur de combinaison de classe « $\oplus$ » spécifie la classe résultat de n'importe quelle fonction binaire ayant comme opérande des classes de sécurité.

– une relation de flux « $\rightarrow$ » entre les paires de sécurité classes A et B signifie que « les informations en classe A sont autorisées à circuler vers la classe B ». Un modèle de flux FM est sûr si et seulement si l'exécution d'une séquence d'opérations ne peut pas produire un flux qui viole la relation « $\rightarrow$ ».

Nous spécifions formellement le problème d'under tainting en utilisant le modèle de flux d'informations de Denning. Mais, nous attribuons une teinte aux objets au lieu d'assigner des classes de sécurité. Ainsi, l'opérateur de combinaison de classe « $\oplus$ » est utilisé dans notre spécification formelle pour combiner les teintes des objets.

Définition syntaxique des connecteurs $\{\Rightarrow, \rightarrow, \leftarrow, \oplus\}$ **:** Nous utilisons la syntaxe suivante pour spécifier formellement le problème d'under tainting : A et B sont deux formules logiques et x et y sont deux variables.

– $A \Rightarrow B$: si A alors B

– $x \rightarrow y$: L'information circule de l'object x à l'object y

– $x \leftarrow y$: la valeur de y est affectée à x

– $Taint(x) \oplus Taint(y)$: spécifie la teinte résultat de combinaisons des teintes.

Définition sémantique de connecteurs $\{\rightarrow, \leftarrow, \oplus\}$ **:**

– Le connecteur $\rightarrow$ est réflexif : Si x est une variable alors $x \rightarrow x$.

– Le connecteur $\rightarrow$ est transitif : x, y et z sont trois variables, si $(x \rightarrow y) \wedge (y \rightarrow z)$ alors $x \rightarrow z$.

– Le connecteur $\leftarrow$ est réflexif : Si x est une variable alors $x \leftarrow x$.

– Le connecteur $\leftarrow$ est transitif : x, y et z sont trois variables, si $(x \leftarrow y) \wedge (y \leftarrow z)$ alors $x \leftarrow z$.

– Les connecteurs $\rightarrow$ et $\leftarrow$ ne sont pas symétriques.

– La relation $\oplus$ est commutative : $Taint(x) \oplus Taint(y) = Taint(y) \oplus Taint(x)$

– La relation $\oplus$ est associative : $Taint(x) \oplus (Taint(y) \oplus Taint(z)) = (Taint(x) \oplus Taint(y)) \oplus Taint(z)$

Definition. Une situation d'under tainting (sous teintage) se produit lorsque x dépend d'une $condition$, on affecte à x une valeur dans la branche conditionnelle et $condition$ est teintée mais x n'est pas teinté. Formellement,

$$\begin{aligned} &\textit{Affectation}(x, y) \wedge Dependance(x, condition) \\ &\qquad \wedge Teinte(condition) \wedge \neg Teinte(x) \end{aligned} \tag{1}$$

où :

– *Affectation*(x, y) affecte à x la valeur de y.

$$\textit{Affectation}(x, y) \stackrel{def}{\equiv} (x \leftarrow y)$$

– $Dependency(x, condition)$ definit un flux d'information de la $condition$ à x si x depend de la $condition$.

$$Dependance(x, condition) \stackrel{def}{\equiv} (condition \rightarrow x)$$

5.2. *Solution formelle de l'under tainting dans les smartphones*

Pour résoudre le problème d'under tainting, nous proposons un ensemble de règles qui définit la politique de teintage permettant de détecter les attaques exploitant les dépendances de contrôles. Grâce à ces règles, toutes les variables auxquelles une valeur

est assignée dans la structure conditionnelle sont teintées que cette branche soit prise ou pas. Nous considérons que le $Contexte\_Teinte$ est la teinte de la $condition$.

– règle 1 : si la valeur de x est modifiée, x dépend de la $condition$ et la branche est prise, nous appliquons la règle suivante pour teinter x.

$$\frac{\textit{Modifier}(x) \wedge Dependance(x, condition) \wedge Branche\_Prise(br, inst\_cond)}{Teinte(x) \leftarrow Contexte\_Teinte \oplus Teinte(inst\_flux\_explicite)}$$

où : Le predicat $Branche\_Prise(br, inst\_cond)$ spécifie que la branche br dans l'instruction conditionnelle est exécutée, et donc un flux explicite qui contient x est exécuté.

Modifier (x, inst_flux_explicite) associe à x le résultat de flux explicite.

$$\textit{Modifier}(x) \stackrel{def}{\equiv} \quad \textit{Affectation(x, inst\_flux\_explicite)}$$

– Règle 2 : si la valeur de y est affectée à x, x dépend de la $condition$ et la branche br dans l'instruction conditionnelle n'est pas prise (x ne dépend que du flux implicite et ne dépend pas du flux explicite), nous appliquons la règle suivante pour teinter x.

$$\frac{\textit{Affectation}(x, y) \wedge Dependance(x, condition) \wedge \neg Branche\_Prise(br, inst\_cond)}{Teinte(x) \leftarrow Teinte(x) \oplus Contexte\_Teinte}$$

Dans ce papier, nous ne reprenons pas la preuve de complétude de ces règles qui est accessible dans (Graa *et al.*, 2013). Dans (Graa *et al.*, 2013), nous avons prouvé la complétude de ces règles. Aussi, nous avons fourni un algorithme correct et complet utilisant ces règles qui permet de résoudre le problème d'under tainting.

5.3. *Solution technique de l'under tainting dans les smartphones*

Pour résoudre le problème d'under tainting, nous sommes intervenu au niveau de l'architecture de TaintDroid. Nous avons implémenté une approche hybride qui combine l'analyse statique et l'analyse dynamique. Nous avons défini et implémenté un module « implicit flow tracking » dans le vérificateur de code Dex de la machine virtuelle Dalvik. Ce module effectue l'analyse statique et vérifie les instructions au moment de l'installation des applications Android. Nous modifions l'interprèteur de la machine virtuelle Dalvik et nous ajoutons les deux règles présentées dans la Section 5.2 pour propager la teinte tout au long des flux de contrôles.

5.3.1. *Analyse statique du Code dex*

Nous effectuons une analyse statique au moment de l'installation des applications Android. Cette analyse utilise des graphes de flot de contrôle qui sont composés des

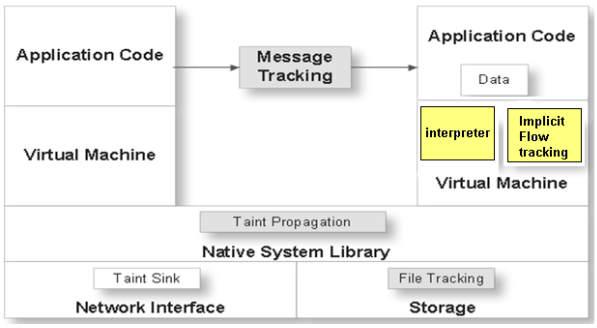


Figure 3. *Architecture modifiée pour gérer les flux implicites dans le système TaintDroid.*

blocs de base et des arêtes. Un bloc de base représente une instruction de contrôle. Nous utilisons « post dominator » pour déterminer la dépendance des différents blocs au bloc de la condition. Les graphes de flot de contrôle sont stockés sous le format graphviz (Research,) dans le dossier de données du smartphone. Les tailles des graphes de flot de contrôle que nous avons obtenus dans nos différents tests sont de l'ordre de 1200 octets.

5.3.2. *Analyse Dynamique des Applications Android*

Nous implémentons l'analyse dynamique au moment de l'exécution en utilisant les informations fournies par l'analyse statique. Nous attribuons un context_taint à chaque bloc de base. Le $Contexte\_Teinte$ contient la teinte de la condition dont dépend le bloc. Nous commençons par les branches qui ne sont pas prises. Nous utilisons l'analyse statique pour déterminer le type et le nombre d'instructions dans ces branches. Ensuite, nous forçons le processeur à exécuter ces instructions et de teinter les variables auxquelles une valeur est affectée en utilisant la deuxième règle de propagation de la teinte. Nous attribuons seulement une teinte aux variables et nous ne modifions pas leurs valeurs. Enfin, nous restaurons le compteur de programme pour pointer vers la première instruction dans la branche qui est prise. Nous attribuons une teinte aux variables modifiées dans cette branche en utilisant la première règle de propagation de la teinte. Nous implémentons les deux règles qui définissent la politique de teintage dans le module « Taint Propagation » de Taintdroid. L'architecture modifiée pour gérer les flux implicites dans le système TaintDroid est illustrée dans la Figure 3.

6. Détection des attaques de contrôle de flux

Nous avons implémenté et testé les trois attaques exploitant les dépendances de contrôles présentées dans la Section 2 en utilisant un smartphone « Nexus One » ayant un système d'exploitation Android 2.3 que nous avons modifié pour suivre les flux

```
W/dalvikvm( 1209): TaintLog: OSNetworkSystem.write(10.35.131.42) received data with tag
 0x2 data=[��
          Mariem Graa]
```

(a)

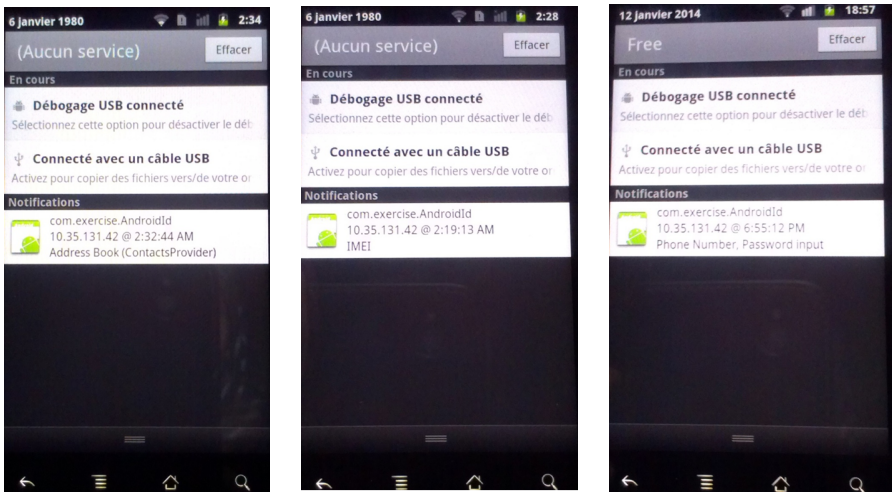
```
W/dalvikvm(  712): TaintLog: OSNetworkSystem.write(10.35.131.42) received data with
 tag 0x400 data=[��354957033679070]
```

(b)

```
W/dalvikvm(  488): TaintLog: OSNetworkSystem.write(10.35.131.42) received data with tag
 0x10008 data=[��
W/dalvikvm(  488): 3627890380]
```

(b)

Figure 4. *Fichier Log des attaques*



(a) (b) (c)

Figure 5. *Notification signalant la fuite des données sensibles*

de contrôles. Nous utilisons l'outil Traceview pour évaluer les performances de ces attaques.

Considérons la première attaque (voir l'Algorithme 1). Nous avons remplacé la donnée privée par le nom du contact de l'utilisateur (« Graa Mariem »). Il compare les caractères de cette donnée avec les symboles de la table Ascii dans la deuxième boucle. La teinte du contact de l'utilisateur est $((u4)0 \times 00000002)$.

La variable x est teintée car elle appartient à la chaîne de caractères X qui est teintée. Ainsi, la condition dans l'instruction $if(x == TabAsc[j])$ est teintée. Notre système propage la teinte dans le flux de contrôle. En appliquant la première règle, Y est teinté et $Taint(Y) = Taint(x == TabAsc[j]) \oplus Taint(Y + TabAsc[j])$. Nous pouvons voir dans le fichier log (Figure 4(a)) que Y est teinté et que la teinte de Y

est égale à la teinte du nom de contact de l'utilisateur. Une notification apparaît (voir Figure 5(a)) pour prévenir l'utilisateur de la fuite de Y qui contient la valeur de la donnée privée. Le premier algorithme est exécuté dans $88ms$ en utilisant Taintdroid modifié pour suivre les flux de contrôles et $36ms$ dans un Android non modifié.

Au niveau de la deuxième attaque (voir l'Algorithme 2), l'attaquant essaye d'obtenir une information secrète X qui est le numéro IMEI du smartphone. La teinte de l'IMEI est $((u4)0 \times 00000400)$. La variable x est teintée car elle appartient à la chaine de caractère X qui est teintée. Le résultat n de la conversion de x en entier est teinté. Ainsi, la condition $(i = 0\ to\ n)$ est teintée. En appliquant la première règle, y est teinté et $Taint(y) = Taint(i = 0\ to\ n) \oplus Taint(y + 1)$. Dans la première boucle, la condition $x \in X$ est teintée. Nous appliquons la première règle, Y est teinté et $Taint(Y) = Taint(x \in X) \oplus Taint(Y + (char)y)$. Ce résultat apparait dans le fichier log illustré dans la Figure 4(b). La fuite de la donnée privée est aussi détectée par notre approche (voir la notification dans Figure 5(b)). Le deuxième algorithme est exécuté dans $101ms$ en utilisant Taintdroid modifié pour suivre les flux de contrôles et de $20ms$ dans un Android non modifié. Le temps d'exécution dans notre approche est plus important parce qu'il inclut le temps de la propagation de la teinte dans les flux de contrôles.

L'attaquant exploite une exception pour lancer la troisième attaque (voir l'Algorithme 3) qui provoque la fuite des données sensibles (numéro de téléphone). Nous avons choisi la division par zéro comme exception arithmétique. Cette exception est teintée et dépend de la condition de la boucle $while$ qui est $y < n$. Cette condition est teintée parce que la variable n qui correspond à la conversion d'un caractère dans $phone\_number$ est teintée. Comme TaintDroid, ne teinte pas les exceptions, nous définissons une teinte d'exception ($Taint\_Exception = ((u4)0 \times 00010000)$). Ensuite, nous propageons la teinte de l'exception dans le bloc catch. Nous appliquons la première règle pour teinter y. On obtient $Taint(y) = Taint(exception) \oplus Taint(y+1)$. Enfin, la chaîne de caractère Y qui contient la donnée privée est teintée et $Taint(Y) = Taint(x \in X) \oplus Taint(Y + (char)y)$. Dans le fichier log illustré dans la Figure 4(c), on peut voir que la teinte de Y est la combinaison de la teinte de l'exception $(((u4)0 \times 00010000))$ et la teinte du numéro de téléphone $(((u4)0 \times 00000008))$. Un message d'alerte apparaît au moment de l'envoi de la donnée sensible (voir la notification dans la Figure 5(c)). L'exécution du troisième algorithme nécessite $1437ms$ en utilisant Taintdroid modifié pour suivre les flux de contrôles et $1385ms$ dans Android non modifié. Cette différence est due à la propagation de la teinte dans le flux de contrôles.

7. Evaluation

Nous avons utilisé CaffeineMark (CORPORATION,) afin de déterminer le « java microbenchmark ». Le premier algorithme présente un score global de 3486 Java instructions exécutées par seconde en utilisant un Android non modifié, et 2893 Java instructions exécutées par seconde en utilisant notre approche. Le deuxième algo-

rithme présente un score global de 3465 en utilisant un Android non modifié et 2893 en utilisant notre approche.

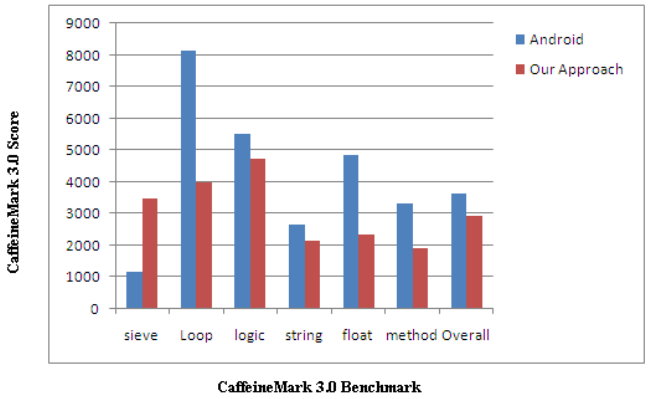


Figure 6. *Microbenchmark de java overhead*

Le troisième algorithme présente un score global de 4241 en utilisant un Android non modifié et 3440 en utilisant approche. Nous avons tester d'autres algorithmes plus complexes. La Figure 6 présente les résultats obtenus. Nous propageons la teinte dans les branches conditionnelles en particulier dans la boucle for et nous ajoutons des instructions dans le processeur pour résoudre le problème d'under tainting ce qui explique le temps d'exécution élevé au niveau de « loop benchmark ». Nous associons une teinte aux résultats des opérations arithmétiques dans les flux explicites et les flux de contrôles. Ainsi, les opérations arithmétiques présentent un temps d'exécution élevé. La différence de « string benchmark » entre un système Android non modifié et notre approche est due à la mémoire supplémentaire requise dans la propagation de la teinte dans les objets string. Nous constatons que le système Android non modifié présente un score global de 3625 Java instructions exécutées par seconde. Notre approche présente un score global de 2937 Java instructions exécutées par seconde. Par conséquent, notre approche crée un overhead de 19% du à la propagation de la teinte dans les flux de contrôles. Cette overhead semble acceptable en comparaison à celui créé par TaintDroid qui est de 14%.

8. Discussion

Dans notre approche, nous associons une teinte à toutes les variables dans la structure conditionnelle. Cela peut causer un problème de faux positifs. Le problème a été abordé dans (Kang *et al.*, 2011) et (Bao *et al.*, 2010). Cependant, ces approches ne proposent pas de solutions. Kang *et al.* (Kang *et al.*, 2011) utilisent une technique de diagnostic pour sélectionner les branches qui pourraient être responsables de l'under tainting et propagent la teinte que seulement tout au long de ces branches afin de

réduire les faux positifs. Cependant DTA++ produit encore des faux positifs. Bao *et al.* (Bao *et al.*, 2010) ne considèrent pas toutes les dépendances de contrôles pour réduire le nombre de faux positifs. Leur mise en oeuvre donne des résultats similaires à DTA++ dans de nombreux cas, et se fonde sur la syntaxe d'une expression de comparaison. En revanche, DTA++ utilise une condition plus générale et plus précise au niveau sémantique, implémentée à l'aide d'une exécution symbolique. Notre approche peut produire des faux positifs, mais elle offre plus de sécurité car toutes les données privées sont teintées. Ainsi, on ne peut pas avoir une fuite des informations sensibles. Nous nous sommes intéressés à résoudre le problème d'under tainting car nous considérons que les faux négatifs sont plus dangereux que les faux positifs étant donné que les faux négatifs peuvent créer des failles de sécurité. Nous avons l'intention de réduire les faux positifs en appliquant des règles expertes. Ce qui permet de réduire aussi les temps d'éxecution trouvés dans la section précédente.

9. Conclusion

Afin de protéger les smartphones des attaques exploitant les dépendances de contrôles, nous avons proposé une approche formelle et technique qui combine l'analyse statique et dynamique. Dans cet article, nous avons présenté des attaques utilisant les dépendances de contrôles qui provoquent la fuite des données sensibles. Nous avons spécifié formellement deux règles qui définissent la politique de teintage pour détecter ces attaques exploitant les dépendances de contrôles. Nous avons montré que notre approche réussit à détecter ces attaques. Ainsi, les applications malveillantes ne peuvent pas contourner le système Android et obtenir des informations sensibles en exploitant les flux de contrôles. Nous avons fait des tests d'évaluation de performance de notre approche qui crée un overhead de 19%. Nous planifions de tester des structures conditionnelles plus complexes (if imbriqué, switch, ...) et d'autres types d'attaques exploitant les dépendances de contrôles. Nous planifions aussi d'affiner notre approche pour réduire le nombre de fausses alarmes.

10. Bibliographie

Bao T., Zheng Y., Lin Z., Zhang X., Xu D., « Strict control dependence and its effect on dynamic information flow analyses », *Proceedings of the 19th international symposium on Software testing and analysis*, ACM, 2010, p. 13–24.

Brown J., Knight Jr T., « A minimal trusted computing base for dynamically ensuring secure information flow », *Project Aries TM-015 (November 2001)*, , 2001.

Bugiel S., Davi L., Dmitrienko A., Heuser S., Sadeghi A.-R., Shastry B., « Practical and lightweight domain isolation on android », *Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices*, ACM, 2011, p. 51–62.

Cavallaro L., Saxena P., Sekar R., « On the limits of information flow techniques for malware analysis and containment », *Detection of Intrusions and Malware, and Vulnerability Assessment*, p. 143–163, Springer, 2008.

Cheng W., Zhao Q., Yu B., Hiroshige S., « Tainttrace : Efficient flow tracing with dynamic binary rewriting », *ISCC'06. Proceedings. 11th IEEE Symposium on*, IEEE, 2006, p. 749–754.

Chin E., Felt A. P., Greenwood K., Wagner D., « Analyzing inter-application communication in Android », *Proceedings of the 9th international conference on Mobile systems, applications, and services*, ACM, 2011, p. 239–252.

Conti M., Nguyen V., Crispo B., « CRePE : Context-related policy enforcement for Android », *Information Security*, , 2011, p. 331–345, Springer.

CORPORATION P. S., « CaffeineMark 3.0 ».

Denning D., « Secure information flow in computer systems », PhD thesis, Purdue University, 1975.

Denning D., « A lattice model of secure information flow », *Communications of the ACM*, vol. 19, n° 5, 1976, p. 236–243, ACM.

Egele M., Kruegel C., Kirda E., Yin H., Song D., « Dynamic spyware analysis », *Usenix Annual Technical Conference*, 2007.

Egele M., Kruegel C., Kirda E., Vigna G., « PiOS : Detecting privacy leaks in iOS applications », *Proceedings of the Network and Distributed System Security Symposium*, 2011.

Egham U., « Gartner Says Worldwide Mobile Phone Sales Grew 35 Percent in Third Quarter 2010 : Smartphone Sales Increased 96 Percent », November 2010.

Enck W., Ongtang M., McDaniel P., « On lightweight mobile phone application certification », *Proceedings of the 16th ACM conference on Computer and communications security*, ACM, 2009, p. 235–245.

Enck W., Gilbert P., Chun B., Cox L., Jung J., McDaniel P., Sheth A., « TaintDroid : An information-flow tracking system for realtime privacy monitoring on smartphones », *Proceedings of the 9th USENIX conference on Operating systems design and implementation*, USENIX Association, 2010, p. 1–6.

Fenton J., « Memoryless subsystem », *Computer Journal*, vol. 17, n° 2, 1974, p. 143–147.

Fuchs A. P., Chaudhuri A., Foster J. S., « SCanDroid : Automated security certification of Android applications », *Manuscript, Univ. of Maryland, http ://www.cs.umd.edu/˜ avik/projects/scandroidascaa*, , 2009, Citeseer.

Graa M., Cuppens-Boulahia N., Cuppens F., Cavalli A., « Formal Characterization of Illegal Control Flow in Android System », *Signal Image Technology & Internet Systems*, IEEE, 2013.

Haselton, « Android Market surpasses 10 billion app downloads, http ://bgr.com/2011/12/06/android-market-surpasses-10-billion-app-downloads-google-kicks-off-0-10-app-sale/ », Decembre 2011.

Hornyack P., Han S., Jung J., Schechter S., Wetherall D., « These aren't the droids you're looking for : retrofitting android to protect data from imperious applications », *Proceedings of the 18th ACM conference on Computer and communications security*, ACM, 2011, p. 639–652.

Hunt A., Thomas D., « Programming Ruby : The Pragmatic Programmer's Guide », *New York : Addison-Wesley Professional.*, vol. 2, 2000.

Kang M., McCamant S., Poosankam P., Song D., « DTA++ : Dynamic taint analysis with targeted control-flow propagation », *Proc. of the 18th Annual Network and Distributed*

System Security Symp. San Diego, CA, 2011.

Laporte, « Les malwares débarquent sur les smartphones, http ://www.igen.fr/iphone/les-malwares-debarquent-sur-les-smartphones-55012% guilf, August 2011.

Nair S., Simpson P., Crispo B., Tanenbaum A., « A virtual machine based information flow control system for policy enforcement », *Electronic Notes in Theoretical Computer Science*, vol. 197, n° 1, 2008, p. 3–16, Elsevier.

Nauman M., Khan S., Zhang X., « Apex : extending android permission model and enforcement with user-defined runtime constraints », *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, ACM, 2010, p. 328–332.

Newsome J., Song D., « Dynamic taint analysis for automatic detection, analysis, and signature generation of exploits on commodity software », Citeseer, 2005.

Ongtang M., Butler K., McDaniel P., « Porscha : Policy oriented secure content handling in Android », *Proceedings of the 26th Annual Computer Security Applications Conference*, ACM, 2010, p. 221–230.

Qin F., Wang C., Li Z., Kim H., Zhou Y., Wu Y., « Lift : A low-overhead practical information flow tracking system for detecting security attacks », *Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture*, IEEE Computer Society, 2006, p. 135–148.

Research A., « Graphviz,http ://www.graphviz.org/ ».

Sarwar G., Mehani O., Boreli R., Kaafar M. A., « On the Effectiveness of Dynamic Taint Analysis for Protecting Against Private Information Leaks on Android-based Devices », 2013.

Song D., Brumley D., Yin H., Caballero J., Jager I., Kang M., Liang Z., Newsome J., Poosankam P., Saxena P., « BitBlaze : A new approach to computer security via binary analysis », *Information Systems Security*, , 2008, p. 1–25, Springer.

Wall L., Christiansen T., Orwant J., *Programming perl*, O'Reilly Media, 2000.

Wilson T., « Many Android Apps Leaking Private Information », July 2011.

Yin H., Song D., Egele M., Kruegel C., Kirda E., « Panorama : capturing system-wide information flow for malware detection and analysis », *Proceedings of the 14th ACM Conference on Computer and Communications Security*, ACM, 2007, p. 116–127.

Session 2a

Impact des *-data sur les systèmes d’information

C-CUBE: Un nouvel opérateur d'agrégation pour les entrepôts de données en colonnes

Khaled Dehdouh — Fadila Bentayeb — Nadia Kabachi — Omar Boussaid

Laboratoire ERIC, Université de Lyon 2

5 avenue Pierre Mendes-France, 69676 Bron Cedex, France

RÉSUMÉ. *Les bases de données orientées colonnes offrent au domaine décisionnel le modèle le plus approprié au stockage des entrepôts de données. Cependant, en l'absence d'opérateurs d'analyse en ligne, le seul moyen, très coûteux, qui existe pour construire des cubes OLAP consiste à utiliser l'opérateur UNION sur des requêtes de regroupement afin d'obtenir l'ensemble des Group By nécessaires au calcul de cube OLAP*[1]*. Pour pallier ce problème, nous proposons dans cet article un nouvel opérateur d'agrégation, baptisé C-CUBE (Columnar-CUBE), qui permet de calculer des cubes de données à partir d'entrepôts de données stockés en colonnes. Nous avons implémenté l'opérateur C-CUBE au sein du SGBD orienté colonnes MonetDB et réalisé des expérimentations sur le benchmark SSBM* [2] *(Star Schema Benchmark). Nous avons ainsi pu montrer que C-CUBE présente des temps de calcul de cubes OLAP jusqu'à 70% moins élevés comparés à l'opérateur CUBE d'Oracle sur un entrepôt de 1To.*

ABSTRACT. *Columnar databases are suitable for data warehouses and multidimensional data structures storage. However, Columnar DBMS have not an appropriate operator for calculating OLAP cubes. In this paper, we propose a new OLAP operator for columnar DBMS, C-CUBE, that allows to calculate OLAP data cubes from columnar oriented-data warehouses. We have then implemented C-CUBE under MonetDB DBMS and curried out some experimentations onto Star Schema Benchmark. The obtained results show that C-CUBE improve the computation time of data cubes up to 70% compared to Oracle CUBE operator.*

MOTS-CLÉS : *Base de données orientée colonnes, Entrepôt de données, Cube OLAP.*

KEYWORDS: *Columnar databases, Data warehouse, OLAP Cube.*

1. On-Line Analytical Processing
2. SSBM: http://www.cs.umb.edu/poneil/StarSchemaB.PDF.

1. Introduction

Une base de donnée orientée colonnes stocke les données d'une table colonne par colonne. Cette technique de stockage permet d'avoir dans un même espace disque les valeurs appartenant à une même colonne, ce qui accélère énormément le temps d'accès à une colonne. Caractérisé par une opération de jointure très performante appelée jointure invisible (Abadi et al., 2008), le stockage en colonnes apparaît comme une solution très intéressante en BI (Business Intelligence) et plus particulièrement pour les entrepôts de données (Matei, 2010). Ces derniers sont dédiés à l'analyse en ligne pour l'aide à la prise de décision (Inmon, 1992). Grâce aux opérateurs OLAP (On- Line Analytical Processing), l'utilisateur peut extraire des cubes de données correspondants à des contextes d'analyse (Han et Kamber, 2006). Le stockage en colonnes est naturellement approprié à la structure de données multidimensionnelles et aux calculs d'agrégats (Stonebraker et al., 2005). Cependant, les fonctionnalités des bases de données orientées colonnes sont limitées. En effet, les SGBD (*Systèmes de Gestion de Bases de Données*) orientés colonnes ne disposent pas d'opérateurs de calcul de cubes OLAP. Toutefois, le cube OLAP peut être calculé avec la méthode naïve en utilisant l'union de requêtes de regroupement (*Group By*) (Dehdouh et al., 2013). Cette méthode de calcul engendre, pour un nombre de dimensions D, l'exécution de 2D sous-requêtes pour calculer les différents agrégats, ce qui augmente considérablement le nombre d'accès à la base de données. Par conséquent, la méthode naïve dégrade les performances du SGBD notamment pour le passage à l'échelle.

L'objectif de cet article est de proposer un opérateur d'agrégation C-CUBE (Columnar-CUBE) pour les SGBD orientés colonnes. Cet opérateur permet de calculer des cubes OLAP à partir d'entrepôts implémentés en colonnes. C-CUBE étend la jointure invisible, utilisée dans les bases de données orientées colonnes, pour prendre en considération toutes les combinaisons de dimensions. De plus, contrairement à la méthode naïve qui sollicite la base de données pour calculer les différents agrégats, C-CUBE exploite une vue (résultat d'une requête d'extraction) définie sur les attributs (dimensions et mesures) nécessaires au calcul du cube OLAP. Cette stratégie lui permet d'éviter le retour aux données de l'entrepôt pour le calcul d'agrégats. Nous avons validé cette technique sur un banc d'essais SSBM (Star Schema Benchmark) (O'Neil et al., 2009), que nous avons stocké au sein du SGBD orienté colonnes MonetDB, et nous avons mené ensuite des expérimentations qui ont permis de montrer, que C-CUBE optimise considérablement le temps de calcul des cubes OLAP comparé à la fois à la méthode naïve et à l'opérateur CUBE d'oracle.

La suite de cet article est organisée de la manière suivante. La section 2 dresse un état de l'art sur les bases de données orientées colonnes. La section 3 présente les différentes notions liées à notre travail. La section 4 est consacrée à la

description détaillée du processus de calcul de cube OLAP et à la présentation de l'opérateur C- CUBE. Nous avons procédé dans la section 5 à l'implémentation de notre opérateur C-CUBE et aux expérimentations d'évaluation. Enfin, nous concluons cet article et présentons quelques perspectives de notre travail dans la section 6.

2. État de l'art

Le stockage des données en colonnes remonte aux années 1970 avec l'utilisation des fichiers transposés et la technique de partitionnement vertical (D.S, 1979). C'est dans les années 1980 que les avantages de la décomposition des modèles de stockage ont été abordés dans la littérature. Ces modèles sont "DSM" (*Decomposition Storage Model*) et NSM (*N-ary Storage Model*) qui sont respectivement les prédécesseurs du stockage orienté colonnes et du stockage orienté lignes (Copeland et Khoshafian, 1985). Ce n'est que dans les années 2000 que le stockage des données en colonnes est apparu comme une alternative au stockage orienté lignes adopté par les SGBD relationnels notamment pour le stockage des bases de données multidimensionnelles (Abadi et al., 2008). Les travaux qui ont été menés dans ce domaine ont permis aux bases de données orientées colonnes de bénéficier indéniablement de meilleures performances relatives à l'opération de jointure, appelée jointure invisible. Rappelons que la jointure invisible est une opération de jointure qui utilise les positions des valeurs et les tables de hachage pour extraire les données qui satisfont les prédicats de la requête (Abadi et al., 2008). En outre, grâce à la technique de la matérialisation tardive (Abadi et al., 2007) et à la technique de compression des données qui agit efficacement sur des valeurs de même type, l'espace nécessaire pour le stockage des données a diminué considérablement (Abadi et al., 2006). Parmi les SGBD orientés colonnes, on peut citer C-Store[3], MonetDB[4] et Vertica[5]. Cependant, même si tout le monde s'accorde à dire que le stockage en colonnes est bien adapté aux données multidimensionnelles et par conséquent au calcul de cubes de données, les SGBD en colonnes ne disposent malheureusement pas d'opérateurs OLAP.

3. Calcul de cubes OLAP dans les entrepôts de données orientés colonnes

Dans cette section, nous définissons le cube OLAP et la méthode naïve qui permet de le calculer.

3. http ://db.csail.mit.edu/projects/cstore/
4. www.monetdb.org
5. http ://www.vertica.org/

3.1. *Cube OLAP*

Un cube de données est une collection de données agrégées et consolidées pour résumer l'information et expliquer la pertinence d'une observation. Il permet de représenter le fait à observer selon plusieurs axes d'observation (dimensions) (Gray et al.,: 1997) qui sont qualifiées de multidimensionnelles, indépendamment de leur support (tables relationnelles ou tableaux multidimensionnels). Le cube de données est exploré à l'aide de nombreuses opérations qui permettent sa manipulation (Rafanelli, 2003). Le calcul de cube permet d'avoir des agrégations au-delà des limites du Group by. Il calcule de façon multidimensionnelle et renvoie dans le cas de calcul d'une somme, des sous-totaux et totaux de toutes les combinaisons possibles. Cela consiste à calculer tous les agrégats suivant tous les niveaux des hiérarchies de toutes les dimensions. Pour un cube à trois dimensions A, B et C, les agrégats calculés concernent les combinaisons sui- vantes : (A, B, C), (A, B, ALL), (A, ALL, C), (ALL, B, C), (A, ALL, ALL), (ALL, B, ALL), (ALL, ALL, C), (ALL, ALL, ALL).

3.2. *La méthode naïve pour le calcul de Cube OLAP*

La méthode naïve pour calculer le cube de données est proposée par (Gray et al., 1997) et consiste à regrouper à l'aide de l'opérateur UNION, une collection de requêtes agrégatives exécutées séparément avec des Group By. Cependant, cette méthode présente l'inconvénient de solliciter de multiples accès à la base de données pour calculer les différents agrégats. En effet, pour un nombre de dimensions D, il y aura 2^D requêtes de regroupement à exécuter. Ce qui dégrade d'avantage les performances du système de gestion de base de données. De ce qui précède, il apparait clairement que la méthode naïve de calcul de cube OLAP n'est pas adaptée aux bases de données volumineuses, car c'est une approche caractérisée par une complexité qui est exponentielle par rapport au nombre de dimensions.

Pour pallier à ce problème et sachant que les SGBD en colonnes ne disposent pas d'opérateurs OLAP, nous proposons un nouvel opérateur d'agrégation, C-CUBE (Columnar-CUBE) permettant de calculer des cubes OLAP à partir d'entrepôts de données stockés en colonnes.

4. C-CUBE : Opérateur CUBE pour les bases de données en colonnes

Nous présentons dans cette section C-CUBE, notre opérateur OLAP pour le calcul de cube dans les entrepôts de données en colonnes. La technique utilisée par l'opérateur C-CUBE que nous proposons consiste à extraire à partir de l'entrepôt, les données qui satisfont tous les prédicats de la requête. Ces données sont regroupées en fonction des colonnes qui représentent les axes d'analyse. Le résultat est donc, une relation R composée des colonnes qui représentent les dimensions

(axes d'analyse) et de (des) colonne(s) représentant la (les) mesure(s) à agréger. La relation R est un résultat intermédiaire qui va servir à calculer l'ensemble des agrégats du cube OLAP. A ce stade, le résultat intermédiaire permet déjà d'obtenir l'agrégation totale et celle en fonction de toutes les colonnes représentant les dimensions. Pour obtenir les autres agrégations des sous-totaux qui composent le cube, chaque dimension au niveau du résultat intermédiaire est hachée avec les valeurs qui la composent pour obtenir des listes de positions, les valeurs de ces listes sont binaires, ils peuvent correspondre à "1" ou à "0", le "1" indique que la valeur de hachage existe à cette position et "0" si non. L'intersection avec un "*ET logique*" des listes de positions des différentes colonnes (dimensions) permet d'avoir un ensemble de listes de positions qui représentent les positions des valeurs de dimensions à combiner et les valeurs de la colonne mesure à agréger et ce, à différent niveau de granularité. Le regroupement de l'ensemble de ces résultats constitue le cube.

Pour détailler les phases d'exécution de cette technique avec un exemple, nous présentons dans la section suivante le banc d'essais décisionnel SSBM

4.1. *Banc d'essais décisionnel SSBM*

Pour décrire et illustrer nos différents propos et contributions, nous présentons dans cette section, le modèle de données en étoile SSBM (Star Schema Benchmark) (O'Neil et al., 2009) présenté dans la figure 1. Nous avons ensuite utilisé cet entrepôt pour évaluer les performances de l'opérateur C-CUBE.

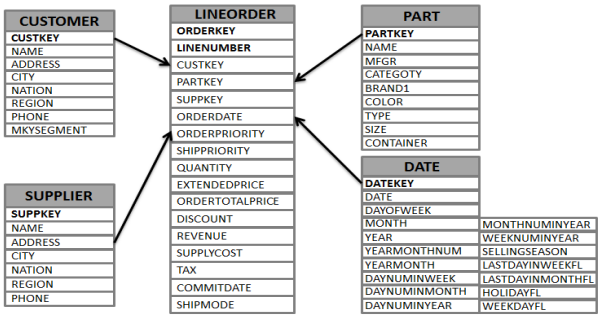


Figure 1. *Modèle en étoile de l'entrepôt de données SSBM*

SSBM est un banc d'essais décisionnel, dérivé du modèle TPC-H[6]. Contrairement à TPC-H, SSBM utilise le schéma en étoile pour permettre d'évaluer les performances de l'entrepôt de données. SSBM est un entrepôt de données qui

6. http:www.tpc.org/tpch/

gère les lignes de commandes en fonction des dimensions PART (produit), SUPPLIER (fournisseur), CUSTOMER (client) et DATE (date). Il est constitué d'une table de faits appelée LINEORDER (lignes de commandes) composée de dix-sept attributs pour renseigner une commande, dont la clé primaire est composée de ORDERKEY et de LINENUMBER et des clés étrangères provenant des tables de dimension. Ce modèle est accompagné d'une charge de requêtes avec de simple *Group by*.

Il est à noter que notre objectif n'est pas d'évaluer SSBM en tant que tel mais de l'utiliser pour évaluer la performance de notre opérateur C-CUBE dans un SGBD orienté colonnes. Pour cela, nous nous abstenons d'utiliser la charge de requêtes qui accompagne SSBM et nous créons de nouvelles requêtes permettant de calculer des cubes OLAP.

Charge de requêtes : Pour évaluer les performances de l'opérateur C-CUBE que nous proposons pour calculer le cube OLAP à partir d'entrepôt stocké en colonnes, nous avons utilisé quatre requêtes de calcul des cubes OLAP. Ces requêtes impliquent graduellement le nombre de dimensions dans le calcul des cubes.

– Requête 1 : C'est une requête qui permet de calculer un cube OLAP à deux dimensions avec des restrictions sur les colonnes, NATION de la dimension CUSTOMER, NATION de la dimension SUPPLIER et YEAR de la dimension DATE. Cette requête calcule à différents niveaux de granularité, la somme des revenus (*sum(revenue)*) en fonction des attributs, CITY de la dimension CUSTOMER et CITY de la dimension SUPPLIER.

– Requête 2 : C'est une requête qui permet de calculer un cube OLAP à trois dimensions avec les mêmes restrictions de la première requête. Elle calcule à différents niveaux de granularité, la somme des revenus (*sum (revenue)*) en fonction des attributs CITY de la dimension CUSTOMER, CITY de la dimension SUPPLIER et YEAR de la dimension DATE.

– Requête 3 : C'est une requête qui permet de calculer un cube OLAP à quatre dimensions avec des restrictions sur les colonnes NATION de la dimension CUSTOMER, NATION de la dimension SUPPLIER, BRAND1 de la dimension PART et YEAR de la dimension DATE. Elle calcule à différents niveaux de granularité, la somme des revenus (*sum (revenue)*) en fonction des attributs CITY de la dimension CUSTOMER, CITY de la dimension SUPPLIER, YEAR de la dimension DATE et BRAND1 de la dimension PART.

– Requête 4 : C'est une requête qui permet de calculer un cube OLAP à cinq dimensions avec les mêmes restrictions de la requête précédente (requête 3). Elle calcule à différents niveaux de granularité, la somme des revenus (*sum (revenue)*) en fonction des attributs CITY de la dimension CUSTOMER, CITY de la dimension SUPPLIER, YEAR et MONTH de la dimension DATE et BRAND1 de la dimension PART.

4.2. *Phases d'exécution de l'opérateur C-CUBE*

L'opérateur C-CUBE calcule le cube OLAP en quatre phases. Pour détailler ces phases, nous illustrons notre explication avec un exemple de la requête 2, citée dans la section 4.1. L'exemple calcule la somme des revenus des ventes des produits livrés par des fournisseurs algériens et commandés par des clients français pendant les années 1996 et 1997.

Première phase : Elle consiste à extraire, à partir de l'entrepôt de données, les données qui satisfont tous les prédicats (filtres). Cette phase permet d'obtenir le résultat intermédiaire pour constituer l'ensemble des parties du cube. Pour réaliser cette phase, les prédicats de la requête sont appliqués séparément sur les dimensions respectives pour obtenir les listes des clés primaires des dimensions qui satisfont les prédicats. Vu que ces clés sont des clés étrangères au niveau de la table de faits, elles sont utilisées pour extraire les listes des positions y afférant au niveau de la table de faits. L'association de ces listes de positions avec un "*ET logique*" génère une seule liste de positions P. Cette dernière représente les positions des valeurs dans la table de faits qui satisfont tous les prédicats de la requête à la fois. L'extraction des valeurs de la table de faits en fonction de P permet d'obtenir le résultat intermédiaire R sous forme d'une vue qui va servir à calculer le cube OLAP. Dans notre exemple, cela revient à sélectionner les clés primaires des villes algériennes (Nation = Algeria) de la dimension SUPPLIER, celles des villes françaises de la dimension CUSTOMER (Nation = France) et celles des années 1996 et 1997 de la dimension DATE (Year in (1996, 1997)). Cette opération donne lieu à trois listes de clés.

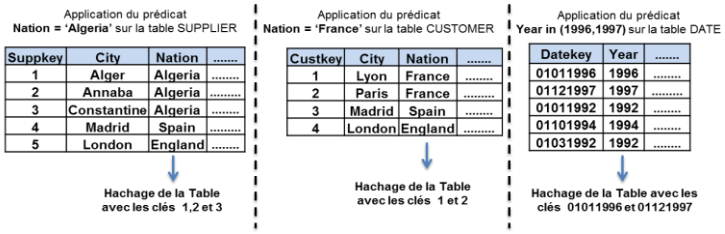


Figure 2. *Extraction des clés des dimensions qui satisfont les prédicats respectives*

Pour obtenir les listes de positions correspondantes à ces clés dans la table de faits, les dimensions Suppkey, Custkey et Orderdate de la table de faits LINEORDER sont hachées sur les trois listes des clés respectives. Cette opération donne lieu à trois listes de positions. Ces dernières, une fois associées génèrent, une liste de positions P qui représente les positions des n-uplets qui satisfont tous les prédicats de la requête. Cette opération donne lieu à un résultat intermédiaire appelé R. A ce stade de l'exécution, les agrégations des sommes des revenus de

(ALL, ALL, ALL) et (Suppkey, Custkey, Orderdate) sont calculées telles que c'est décrit dans la figure 3.

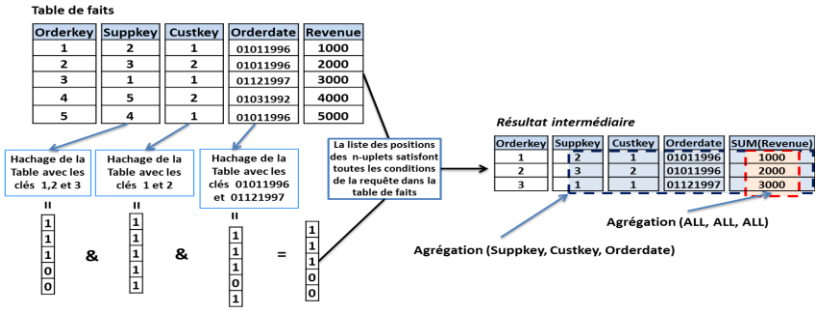


Figure 3. *Extraction des données qui satisfont tous les prédicats de la requête et construction du résultat intermédiaire (relation R)*

Deuxième phase : Dans cette phase, chaque colonne du résultat intermédiaire R représentant une dimension est hachée avec les valeurs qui la composent pour obtenir les listes des positions de ces valeurs. En effet, la dimension Suppkey du résultat intermédiaire est hachée sur les valeurs (2, 3 et 1) donnant lieu respectivement à trois listes de positions $P_{Suppkey(2)}$, $P_{Suppkey(3)}$ et $P_{Suppkey(1)}$. Custkey est hachée sur les valeurs (1 et 2) donnant lieu à $P_{Custkey(1)}$ et $P_{Custkey(2)}$ et enfin, Orderdate est hachée sur (01011996 et 01121997) donnant lieu à $P_{Orderdate(1996)}$ et $P_{Orderdate(1997)}$. A ce stade de l'exécution, ces listes de positions permettent d'agréger les valeurs de la colonne représentant la mesure, et ce, pour chaque dimension séparément. En effet, elles fournissent les agrégations des sommes des revenus suivantes : (Suppkey, ALL, ALL), (ALL, Custkey, ALL) et (ALL, ALL, Orderdate).

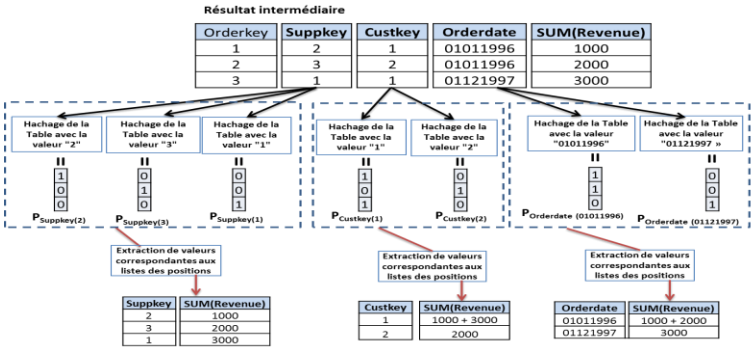


Figure 4. *Construction des listes de positions et calcul d'agrégats pour chaque dimension du résultat intermédiaire.*

Troisième phase : Au niveau de cette phase, les listes de positions des dimensions au niveau de R sont associées via l'opérateur "*ET logique*". Cette opération permet d'identifier les valeurs des dimensions à combiner et les valeurs de la mesure à agréger qui correspondent aux différentes combinaisons. Dans ce cas, pour identifier les différentes combinaisons possibles entre les dimensions Suppkey et Custkey, les listes de positions $P_{Suppkey(2)}$, $P_{Suppkey(3)}$ et $P_{Suppkey(1)}$ sont associées avec $P_{Custkey(1)}$ et $P_{Custkey(2)}$ via l'opérateur "*ET logique*". Les résultats sont les trois listes $P_{Suppkey(2)-Custkey(1)}$, $P_{Suppkey(3)-Custkey(2)}$ et $P_{Suppkey(1)-Custkey(1)}$. Ces listes permettent d'extraire les valeurs des dimensions Suppkey, Custkey et de mesure Revenue à agréger. Ces opérations fournissent les agrégations des sommes des revenus suivantes : (Suppkey, Custkey, ALL), (Suppkey, ALL, Orderdate) et (ALL, Custkey, Orderdate).

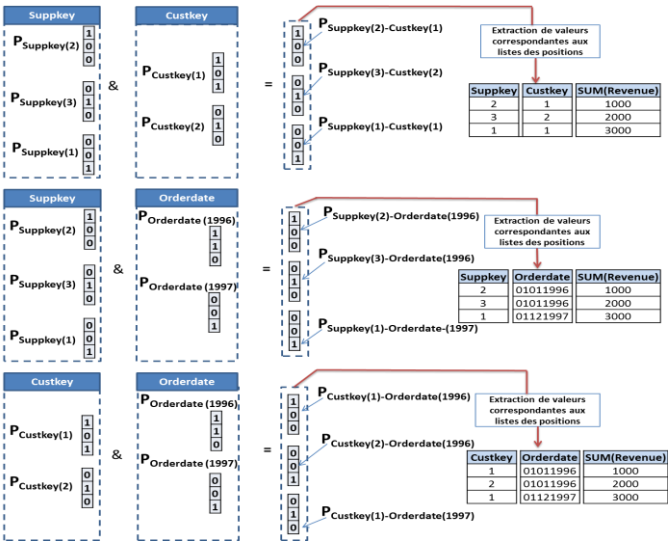


Figure 5. *Calcul d'agrégat pour les différentes combinaisons des listes de positions*

Quatrième phase : Elle consiste à regrouper toutes les combinaisons et les agrégats réalisés. Ensuite, elle extrait les valeurs à afficher correspondantes aux clés des dimensions. En effet, la construction de toutes les combinaisons pour calculer les différents agrégats qui constituent le cube OLAP a été réalisée avec les clés de dimensions. Dans le cas de notre exemple, les clés (2, 3, 1) de la dimension Suppkey correspondent aux valeurs (Annaba, Constantine, Alger) de la dimension SUPPLIER, les clés (1, 2) de la dimension Custkey correspondent aux valeurs (Lyon, Paris) de la dimension CUSTOMER et enfin les clés (01011996, 01121997) correspondent aux valeurs (1996, 1997) de la dimension DATE. Par conséquent,

le regroupement des sous-résultats (totaux et sous-totaux) des trois phases précédentes permet de calculer le cube OLAP représenté dans la figure 6.

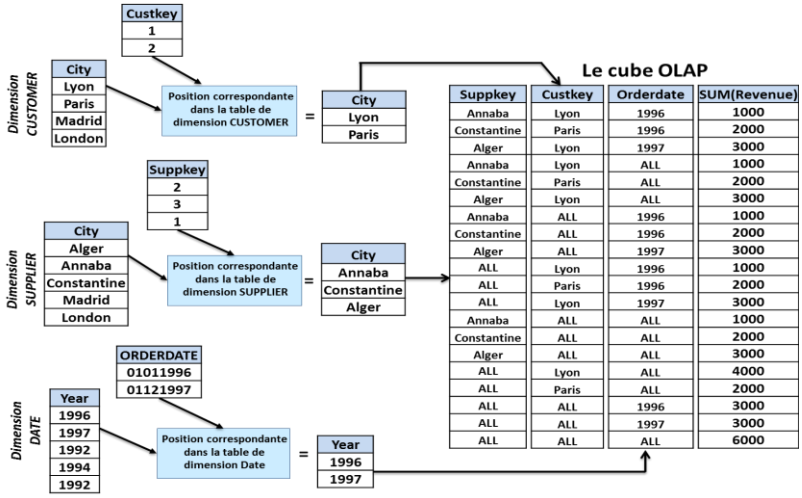


Figure 6. *Regroupement des agrégats (totaux et sous-totaux) et extraction des valeurs à afficher, correspondantes aux clés de dimensions.*

Noter bien que la quasi-totalité des traitements ont été effectués avec des clés et des listes de positions. Ce procédé est très avantageux aux traitements au niveau de la mémoire car il offre la possibilité de réaliser le maximum des opérations, sans solliciter pour autant les accès au disque, ce qui permet de réduire considérablement le flux d'entrées/sorties.

5. Implémentation et expérimentations

5.1. *Implémentation*

Pour valider notre méthode de calcul de cube OLAP à partir d'entrepôt de données orienté colonnes, nous avons implémenté l'opérateur C-CUBE en JAVA. Par ailleurs, pour mener nous expérimentations, nous avons implémenté le banc d'essais décisionnel SSBM, décrit dans la section 4.1, sous le SGBD MonetDB (Idreos et al., 2012). Le choix de ce dernier est motivé par le fait qu'il est orienté colonnes et en sources libres (open sources). L'environnement de tests que nous avons utilisé consiste en une machine intel-Core TMi3-3220 CPU@3.30 GHZ avec une mémoire RAM de 4Go. Cette machine fonctionne avec le système d'exploitation Microsoft Windows 7 de 64bits.

5.2. *Expérimentations*

Dans cette partie de l'article, nous avons évalué les performances de l'opérateur C-CUBE en matière de temps de calcul du cube OLAP sur l'entrepôt de données SSBM selon deux architectures. La première est relationnelle orientée lignes avec le SGBD Oracle 11g. La deuxième est relationnelle orientée colonnes avec le SGBD MonetDB. Les deux implantations nous permettent de comparer le temps d'exécution des requêtes de construction des cubes OLAP selon l'opérateur C-CUBE, la méthode naïve et l'opérateur CUBE d'oracle. Nous avons ensuite mené deux expérimentations. La première évalue le temps de calcul des cubes OLAP avec un nombre de dimensions qui augmente graduellement. La deuxième évalue le gain en matière de temps de calcul de cube OLAP entre l'opérateur C-CUBE et l'opérateur CUBE en faisant varier la taille de l'entrepôt.

5.2.1. *Calcul des cubes OLAP*

L'objectif de cette expérimentation est d'observer le comportement de notre opérateur C-CUBE face aux variations du nombre de dimensions. Nous avons fixé la taille de l'entrepôt de données à 50 Go. Ensuite, nous avons comparé la performance de C-CUBE avec la méthode naïve de l'approche orientée colonnes et l'opérateur CUBE du SGBD Oracle. Pour cela, nous avons exécuté les quatre requêtes présentées dans la section 4.1. Ces requêtes calculent des cubes OLAP avec un nombre de dimensions qui augmente progressivement. Les résultats que nous avons obtenus sont présentés dans la figure 7.

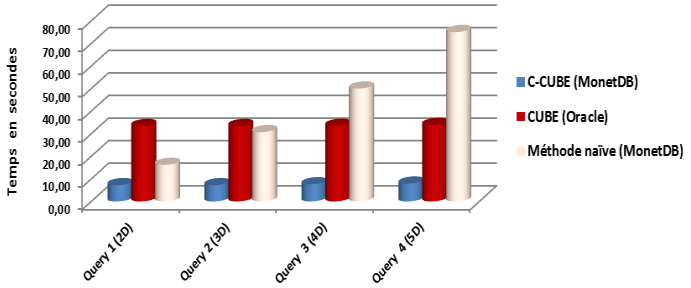


Figure 7. *Résultats de calcul des cubes OLAP à 2, 3, 4 et 5 dimensions*

Nous constatons que les temps de calcul des cubes OLAP avec la méthode naïve varient en moyenne entre 16.2 et 75 secondes. Ces temps de calcul augmentent en fonction du nombre de dimensions. En effet, le temps nécessaire pour calculer un cube à deux dimensions est de 16.2, et pour un cube à trois dimensions est de 30.7 secondes, cela représente presque le double. Au-delà de trois dimensions, cette méthode enregistre de mauvais résultats par rapport aux autres méthodes. Ce

résultat est expliqué par le fait que pour deux dimensions, le système exécute quatre (2^2) requêtes agrégatives, souvent avec des jointures répétitives et des regroupements de résultats. Par contre, avec trois dimensions, il exécute huit (2^3) requêtes agrégatives, ce qui représente pratiquement le double. Cependant, l'augmentation du nombre de dimension implique des accès disques importants pour extraire les données de l'entrepôt, cela engendre au niveau de la mémoire une gestion supplémentaire des résultats intermédiaires. Par conséquent, il en découle une saturation de la mémoire, cette dernière sollicite le disque pour gérer les résultats intermédiaires. De ce fait, il est clair que ce procédé augmente considérablement le coût d'entrées/sorties qui se traduit par un temps d'exécution plus élevé et une dégradation des performances du système.

En revanche, nous constatons une légère variation des temps d'exécution des requêtes générant les cubes OLAP, avec les opérateurs C-CUBE et CUBE. Cependant, C-CUBE affiche une meilleure performance que CUBE, en effet, l'opérateur CUBE enregistre des temps entre 33.4 et 33.9 secondes, alors que l'opérateur C-CUBE enregistre des temps entre 7.2 et 7.9 secondes.

L'avantage de C-CUBE réside dans l'utilisation et l'exploitation des listes de positions pour le calcul des cubes OLAP. Ces listes occupent peu d'espace mémoire, elles conviennent parfaitement à un traitement au niveau de la mémoire sans pour autant solliciter de multiples accès au disque. En effet, l'augmentation du nombre de dimensions dans le calcul de cube de données se traduit techniquement par la création et la manipulation des vecteurs de bits qui représentent des listes des positions des valeurs. Ce procédé n'impacte pas lourdement la mémoire. De plus, les combinaisons sont réalisées avec des listes (vecteur de bits composé de "1" ou "0") et non pas avec des valeurs, ces dernières ne sont extraites qu'après avoir construit la liste des positions y afférente à une combinaison.

Eu égard des résultats obtenus, nous avons constaté que l'opérateur C-CUBE que nous avons proposé optimise considérablement le temps de calcul de cube OLAP. Pour cela, il était très intéressant d'évaluer le comportement face à un passage à l'échelle en augmentant la taille de l'entrepôt jusqu'à 1 To.

5.2.2. *Calcul des cubes OLAP avec un passage à l'échelle*

L'objectif de cette expérimentation est d'évaluer en pourcentage le gain en matière de temps de calcul de cube OLAP offert par l'opérateur C-CUBE, par rapport à l'opérateur CUBE des SGBDR orientés lignes. Pour cela, nous avons confronté l'opérateur C-CUBE au passage à l'échelle. L'expérimentation consiste à évaluer le temps d'exécution de la requête 2 de la section 4.1 qui calcule un cube OLAP à trois dimensions, avec des échantillons de données qui augmentent progressivement, allant de 100 G0 jusqu'à 1 To. Les résultats que nous avons obtenus sont présentés dans la figure 8.

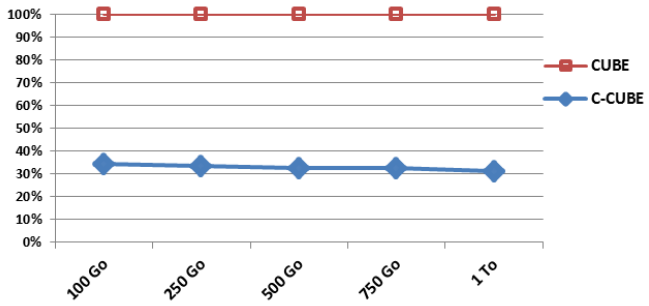


Figure 8. *Résultat de calcul des cubes OLAP avec un passage à l'échelle*

Cette figure représente la tendance en pourcentage des temps de calcul des cubes OLAP entre les opérateurs C-CUBE et CUBE dans des différentes tailles de l'entrepôt de données.

Nous constatons que la courbe représentant le temps de calcul des cubes OLAP avec l'opérateur C-CUBE présente de meilleurs résultats et ce, quelle que soit la taille de l'entrepôt. En effet, L'écart des résultats des temps de calcul des cubes OLAP des deux opérateurs varie en générale entre 65% et 70% pour une taille de l'entrepôt allant de 100 Go à 1 To. Cet écart en faveur de l'opérateur C-CUBE démontre clairement son avantage.

La performance de l'opérateur C-CUBE s'explique par le fait que le système exploite une vue (résultat d'une requête d'extraction) définie sur les attributs (dimensions et mesures) nécessaires au calcul de cube OLAP. Cette stratégie lui permet de réaliser les différentes combinaisons et le calcul d'agrégats au niveau de la mémoire et d'éviter le retour aux données de l'entrepôt. En effet, Cela est possible grâce à l'utilisation de la position de la valeur au lieu de la valeur elle-même. Les positions de valeurs représentées par les listes de vecteurs n'occupent pas beaucoup d'espace au niveau de la mémoire. Ce qui offre la possibilité d'effectuer plusieurs traitements au niveau de la mémoire et diminue par conséquent considérablement le flux d'entrée et sortie.

Au final, les expérimentations que nous avons réalisées démontrent clairement que l'opérateur C-CUBE que nous avons proposé pour calculer le cube OLAP dans les bases de données orientées colonnes est performant par rapport à l'opérateur CUBE d'Oracle.

6. Conclusion

L'intérêt majeur de ce travail est d'étendre l'utilisation des bases de données en colonnes au domaine décisionnel. Dans ce contexte, nous avons proposé dans cet article un opérateur de calcul du cube OLAP, baptisé C-CUBE. L'avantage de cet opérateur est qu'il s'appuie sur le principe de la jointure invisible qui est exploitée ici pour calculer de façon efficace les agrégats du cube OLAP. En effet, à l'instar de la technique utilisée dans la jointure invisible, C-CUBE manipule des listes des positions des valeurs et des tables de hachage qui contiennent des clés des différentes dimensions invoquées dans la requête. Cette manière de procéder réduit considérablement le flux d'entrée et sortie. L'implémentation de cet opérateur au sein du SGBD orienté colonnes MonetDB, et les expérimentations que nous avons menées sur l'entrepôt de données relationnel SSBM ont montré clairement la performance de notre opérateur comparée à celle de l'opérateur CUBE d'Oracle.

De manière plus générale, l'opérateur C-CUBE peut être appliqué sur tout SGBD orienté colonnes et peut être généralisé même aux SGBD orientées colonnes non relationnels (NoSQL[7]). C'est dans ce contexte que ce travail ouvre plusieurs perspectives de recherche intéressantes. L'une des pistes de recherche consiste à adapter l'opérateur C-CUBE aux calculs des cubes OLAP à partir des entrepôts de données NoSQL qui gèrent des Big Data à grande échelle (Jerzy, 2012).

7. Bibliographies

Abadi D., Madden S., Ferreira M., « Integrating compression and execution in column oriented database systems », *Special Interest Group on Management of Data Conference*, 2006, p. 671-682.

Abadi D., Madden S., Hachem N., « Column-stores vs. row-stores: how different are they really? », *International Conference on Management of Data*, p2008, 967-980.

Copeland G., Khoshafian S., « A decomposition storage model », *Special Interest Group on Management Of Data Record*, 1985, p. 268-279.

Dehdouh K., Bentayeb F., Kabachi N., « Performances de requêtes OLAP dans les bases de données en colonnes », *Conférence sur les Avancées des Systèmes Décisionnels*, 2013, p. 439-444.

Batory, D., « On searching transposed files », *Association for Computing Machinery*, 1979, p.531-544.

Gray J., Chaudhuri S., Bosworth A., Layman A., Reichart D., Venkatrao M., Pellow F., Pirahesh H., « Data cube : A relational aggregation operator generalizing group-by,

7. Not Only Sql

cross-tab, and sub totals », *Journal of Data Mining and Knowledge Discovery*, 1997, p.29-53.

Han J., Kamber M., Data mining : concepts and techniques, Morgan Kaufmann Publishers Inc, San Francisco, CA, USA, 2006.

Idreos v., Groffen F., Nes v., Manegold S., Mullender K., Sjoerd K., Kersten v., « MonetDB : Two Decades of Research in Column-oriented Database Architectures », *Journal IEEE Data Engineering. Bull*, 2012, p.40-45.

Inmon W., « Building the Data Warehouse », 1992.

Jerzy D., « Business Intelligence and NoSQL Databases », *Information Systems in Management*, 2012, p.25-37.

Matei G., « Column-Oriented Databases, an Alternative for Analytical Environment », *Database Systems Journal*, 2010, p. 3-16.

O'Neil P., O'Neil B., Chen X., « The Star Schema Benchmark (SSBM) », 2009. http ://www.cs.umb.edu/p˜oneil/StarSchemaB.PDF.

Rafanelli M., « Operators for Multidimensional Aggregate Data », *Multidimensional Databases: Problems and Solutions, IGI Publishing Group*, 2003, p. 116-165.

Stonebraker v., Abadi D., Batkin A., Chen X., Cherniack M., Ferreira M., Lau E., Lin v., Madden S., O'Neil E., O'Neil P., Rasin A., Tran N., Zdonik S., « C-store : a column-oriented DBMS », *Proceedings of the 31st International Conference on Very Large Data Bases*, 2005, p. 553-564.

PF-ETL : vers l'intégration de données massives dans les fonctionnalités d'ETL

Mahfoud Bala[1], Omar Boussaid[2], Zaia Alimazighi[3], Fadila Bentayeb[2]

1. LRDSI, Université Saad Dahleb de Blida
BP 270, route de Soumaa, Blida, Algérie
mahfoud.bala@gmail.com

2. ERIC, Université Lumière, Lyon 2
5, Avenue Pierre Mendès, 69676 Bron Cedex – France
{omar.boussaid, fadila.bentayeb}@univ-lyon2.fr

3. LSI, Université des Sciences et des Technologies Houari Boumédiène, Alger
BP 32, EL ALIA 16111 Bab Ezzouar, Alger, Algérie
zalimazighi@usthb.dz

RESUME. *Un processus ETL (Extracting-Transforming-Loading) est responsable d'extraire des données à partir de sources hétérogènes, les transformer et enfin les charger dans un entrepôt de données. Les nouvelles technologies, particulièrement Internet et le Web 2.0, générant des données à une vitesse croissante, ont mis les systèmes d'information (SI) face au défi du Big Data. Ces données sont caractérisées par, en plus de leur volumétrie et la vitesse avec laquelle elles sont générées, une hétérogénéité plus importante suite à l'émergence de nouvelles structures de données. Les systèmes d'intégration et l'ETL en particulier doivent être repensés et adaptés afin de faire face à l'impact des Big Data. Dans ce contexte et pour mieux gérer l'intégration de données massives, nous proposons une nouvelle approche du processus ETL pour lequel nous définissons des fonctionnalités pouvant s'exécuter sur un cluster selon le modèle MapReduce (MR).*

ABSTRACT. *ETL process (Extracting, Transforming, Loading) is responsible for extracting data from heterogeneous sources, transforming and finally loading them into a data warehouse. New technologies, particularly Internet and Web 2.0, generating data at an increasing rate, put the information systems (IS) face to the challenge of Big Data. These data are characterized by, in addition to their excessive sizes and speed with which they are generated, greater heterogeneity due to the emergence of new data structures. Integration systems and ETL in particular should be revisited and adapted to cope with the impact of Big Data. In this context and to better manage the integration of Big data, we propose a new approach to ETL process for which we define features that can be run easily on a cluster with MapReduce (MR) model.*

MOTS-CLES : *ETL, Données massives, Entrepôts de données, MapReduce, Cluster*

KEYWORDS: *ETL, Big Data, Data warehouse, MapReduce, Cluster*

1. Introduction

L'utilisation à grande échelle d'Internet, du Web 2.0 et des réseaux sociaux produit, instantanément, des volumes non habituels de données. Des *jobs MapReduce* exécutés en continu sur des *clusters* de *Google* traitent plus de vingt *PetaBytes* de données par jour (cf. Ghemawat et Dean, 2008). Cette explosion de données est une opportunité dans l'émergence de nouvelles applications métiers mais en même temps problématique face aux capacités limitées des machines et des applications. Ces données massives, connues aujourd'hui sous le nom de *Big Data,* sont caractérisées par les trois V (cf. Mohanty et *al.*, 2013) : Volume qui implique la quantité des données qui va au-delà des unités habituelles, la Vélocité qui implique la rapidité avec laquelle ces données se génèrent et doivent être traitées et enfin la Variété qui implique la diversité des formats et structures. En parallèle à l'émergence du *Big Data*, de nouveaux paradigmes ont vu le jour tels que le *Cloud Computing* (cf. Barrie Sosinsky, 2011) et *MapReduce (MR)* (cf. Dean et Ghemawat, 2004). Par ailleurs, de nouveaux modèles de données non-relationnelles émergent, appelés modèles *NoSQL (Not Only SQL)* (cf. Han et *al.,* 2011).

L'objectif de cet article est d'apporter des solutions aux problèmes posés par les *Big Data* dans un environnement décisionnel. Nous nous intéressons en particulier à l'intégration de données massives dans un entrepôt de données. Nous proposons alors un processus d'ETL parallèle, appelé *PF-ETL (Parallel Functionality-ETL),* composé d'un ensemble de fonctionnalités selon le paradigme *MR*. Les solutions proposées par la communauté, dans ce contexte, consistent à instancier le processus ETL sur plusieurs nœuds d'un cluster. Chacune des instances traite une partition des données sources de façon parallèle afin d'améliorer les performances du processus ETL. A notre connaissance, *PF-ETL*, constitue une nouvelle approche dans le domaine de l'intégration de données. Nous définissons tout d'abord un processus ETL à un niveau très fin en le décrivant par un ensemble de fonctionnalités de base. Celles-ci sont conçues avec le modèle *MR* de manière à pouvoir les instancier ainsi que le processus ETL sur les différents nœuds d'un cluster. *PF-ETL* permet alors un parallélisme selon le modèle *MR* à deux niveaux (1) niveau fonctionnalité d'ETL et (2) niveau processus ETL ; ce qui permettra d'améliorer davantage les performances de l'ETL face aux *Big Data*. Plusieurs travaux sur l'ETL existent dans la littérature. Nous proposons une classification de ces travaux, sous forme de familles d'approches, selon des critères relevant de la parallélisation et comparons ensuite notre proposition par rapport à ces approches. Pour valider notre processus *PF-ETL*, nous avons développé un prototype logiciel et mené des expérimentations.

Le présent article est organisé de la manière suivante. Dans la section 2, nous définissons le processus ETL. Dans la section 3, nous exposons un état de l'art sur les travaux dans le domaine de l'ETL. Nous proposons ensuite dans la section 4 notre classification des processus ETL proposés dans la littérature sous forme d'approches selon le critère de parallélisation. La section 5 est consacrée à notre processus ETL parallèle, *PF-ETL*. Nous développons dans la section 6 les différentes expérimentations que nous avons menées et présentons nos résultats. Nous concluons et présentons nos travaux futurs de recherche dans la section 7.

2. Processus ETL (*Extracting-Transforming-Loading*)

Le processus ETL est un ensemble de tâches classées en trois catégories. Les tâches d'extraction (E) permettent de se connecter sur diverses sources pour extraire les données les plus pertinentes. Les tâches de transformation (T) préparent les données en vue de leur affecter des propriétés en termes de qualité, de format et de pertinence. Enfin, les tâches de chargement (L), basées sur un schéma de mappage, permettent de charger les données dans l'entrepôt de données (*DW : Data Warehouse*). La couche inférieure de la figure 1 présente la partie statique qui montre les *sources* de données, le *Data Staging Area (DSA)* vers lequel sont rapatriées les données pour être préparées et le *DW* représentant la destination finale des données. Dans la couche supérieure, sont représentées les trois phases d'ETL à savoir *extraction*, *transformation* et *chargement*.

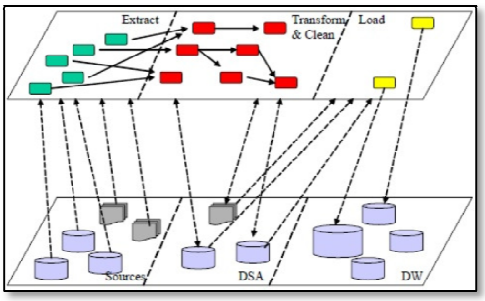


Figure 1. Environnement d'un processus ETL (cf. Vassiliadis et al., 2002)

Nous assistons, depuis plus d'une décennie, à une évolution ayant affecté l'aspect statique du processus ETL suite à l'émergence du *Big Data* et aux nouvelles technologies de stockage et son aspect dynamique suite à l'avènement du paradigme *MR* et des environnements *cloud computing*. L'ETL devra s'adapter à cette évolution en intégrant ces nouvelles technologies qui impactent, bien évidemment, son architecture et son processus mais tout en préservant sa vocation qui est l'intégration des données pour des fins d'analyse.

3. Etat de l'art

L'une des premières contributions dans le domaine de l'ETL est celle de Vassiliadis et *al.* (2002). Il s'agit d'une approche de modélisation basée sur un formalisme graphique non standard ; *ARKTOS II* étant le prototype mis en œuvre. Trujillo et Luján-Mora (2003) se sont intéressés, eux aussi, à la modélisation de l'ETL dans une approche plus globale basée sur des choix standards, en particulier un formalisme qui s'appuie sur une extension de la notation *UML* (*Unified Modeling Language*). El Akkaoui et Zemányi (2009) ont adopté, quant à eux, la notation standard *BPMN* (*Business Process Model and Notation*) dédiée à la modélisation

des processus métier. Ce travail a été poursuivi par El Akkaoui et *al.* (2011) en proposant un framework de modélisation basé sur un métamodèle dans une architecture *MDD (Model Driven Development).*

Le travail de Liu et *al*. (2011) s'intéresse aux performances de l'ETL face au *Big Data* et adopte le modèle *MR*. Cette approche a été implémentée dans un prototype appelé *ETLMR*, version *MR* du prototype *PygramETL* (cf. Thomson et Pederson, 2009a). Bala et Alimazighi (2013) ont proposé une modélisation de l'ETL pour le *Big Data* selon le paradigme *MR* en adoptant le formalisme de Vassiliadis et *al.* (2002) enrichi par des notations graphiques pour modéliser les spécificités du modèle *MR*. Récemment, Oliveira et Belo (2013) ont proposé une approche de modélisation qui consiste en une vue résumée du processus ETL et adopte le modèle *Reo* (cf. Arbab, 2003). Nous considérons que cette contribution pourrait être intéressante mais n'est pas suffisamment mature et mérite une personnalisation du modèle *Reo* pour prendre en charge les spécificités de l'ETL. Très récemment, les expérimentations de Misra et *al.* (2013) montrent que les solutions ETL basées sur des frameworks *MR* open source tel que *Hadoop* sont très performantes et moins coûteuses par rapport aux outils ETL commercialisés.

Il apparait à travers cet état de l'art que les contributions portent essentiellement sur la modélisation du processus ETL et l'amélioration de ses performances. Concernant la modélisation du processus ETL, l'approche de Vassiliadis et *al.* (2002) propose un schéma qui intègre la plupart des aspects d'ETL à un niveau très fin, autrement dit l'attribut. Cependant, elle ne prévoit pas de découpage de l'ETL en sous-processus pour une meilleure lisibilité du modèle, chose qui est proposée, dans une démarche *top-down,* par Trujillo et Luján-Mora (2003) et El Akkaoui et Zemányi (2009). Le travail de Bala et Alimazighi (2013) est une contribution de modélisation qui tente d'intégrer les *Big Data* dans un modèle *MR*. Quant aux travaux ayant pour objectif l'amélioration des performances, ceux-ci ont adopté le modèle *MR* qui accélère considérablement l'ETL face aux *Big Data* (cf. Misra et *al.,* 2013). Contrairement à Liu et *al.* (2011), Misra et *al.* (2013) considèrent la phase d'extraction (E) de l'ETL coûteuse et l'ont traitée avec le modèle *MR*. Nous considérons, à l'issue de cet état de l'art, que l'ETL face à l'évolution des données ainsi qu'à l'émergence de nouveaux paradigmes est une problématique qui reste d'actualité.

4. Classification des travaux sur le processus ETL

Nous proposons, dans cette section, une classification des travaux sur l'ETL sous forme de deux approches en se basant sur la parallélisation de celui-ci : « approche classique » et « approche basée sur le modèle *MR* ».

4.1. Processus ETL classique

Dans ce papier, nous considérons que le processus ETL est classique lorsque celui-ci s'exécute sur un serveur en une seule instance (une seule exécution en même temps) et où les données sont de taille modérée.

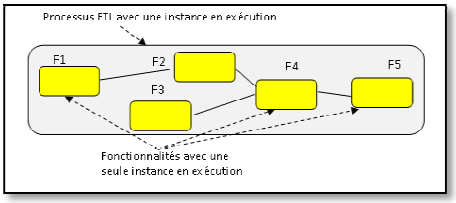


Figure 2. Processus ETL classique

Dans ce contexte, seules les fonctionnalités d'ETL indépendantes peuvent s'exécuter en parallèle. Une fonctionnalité d'ETL, telles que *Changing Data Capture* (*CDC*), *Surrogate Key* (*SK*), *Slowly Changing Dimension (SCD)*, *Surrogate Key Pipeline (SKP)*, est une fonction de base qui prend en charge un aspect particulier dans un processus ETL. La fonctionnalité *CDC*, par exemple, permet dans la phase d'extraction d'un processus ETL d'identifier, dans les systèmes sources, les tuples affectés par des changements afin de les capturer et les considérer dans le rafraichissement du *DW*. Dans la figure 2, nous remarquons que les fonctionnalités *F1* et *F3* ou *F2* et *F3* peuvent s'exécuter en parallèle.

4.2. Processus ETL basé sur le modèle MapReduce (MR)

Le processus ETL, avec un schéma classique, ne pourra pas faire face à l'intégration de données massives. Le paradigme *MR* permet de partitionner de gros volumes de données dont chaque partition sera soumise à une instance du processus ETL.

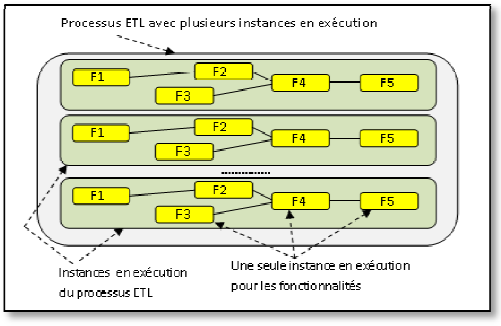


Figure 3. Processus ETL basé sur le modèle MR

Comme le montre la figure 3, plusieurs instances du processus ETL s'exécutent en parallèle où chacune d'elles traitera les données de sa partition dans une phase appelée *Map*. Les résultats partiels seront fusionnés et agrégés, dans une phase *Reduce,* pour obtenir des données prêtes à être chargées dans le *DW*.

4.3. Classification des travaux sur l'ETL

Les travaux présentés dans le tableau 1 portent sur la modélisation ou sur l'optimisation du processus ETL mais interviennent tous à un niveau processus. Les fonctionnalités d'ETL étant les fonctions de base autour desquelles s'articule ce processus. Elles méritent une étude approfondie, notamment les plus courantes d'entre-elles, afin de garantir, à un niveau très fin, une robustesse, fiabilité et optimisation du processus ETL.

Tableau 1. Classification des travaux sur le processus ETL

Contributions	Objectif	Classification
Vassiliadis et *al.* (2002)	Modélisation	Approche Classique
Trujillo et Luján-Mora (2003)	Modélisation	Approche Classique
El Akkaoui et Zemányi (2009)	Modélisation	Approche Classique
Liu et *al.* (2011)	Performances	Approche basée sur *MR*
Bala et Alimazighi (2013)	Modélisation	Approche basée sur *MR*
Oliveira et Belo (2013)	Modélisation	Approche Classique
Misra et *al.* (2013)	Performances	Approche basée sur *MR*
Bala et *al.* (2014)	Performances	Approche basée sur *MR*

5. *PF-ETL* : ETL parallèle basé sur des fonctionnalités

Pour prendre en charge les problèmes posés par les *Big Data*, nous proposons dans cet article un processus ETL, baptisé *PF-ETL (Parallel Functionality-ETL)*, décrit par un ensemble de fonctionnalités où chacune peut s'exécuter en plusieurs instances parallèles (plusieurs exécutions simultanées de la même fonctionnalité) afin de tirer profit des nouveaux environnements informatiques. Notre processus *PF-ETL* s'inscrit dans un objectif d'amélioration de performances (Tableau 1). Parmi les caractéristiques des *Big Data*, nous nous sommes intéressés en particulier au volume et à la vélocité des données. La variété en termes de format n'est pas traitée dans ce travail. Pour intégrer les *Big Data* dans un *DW*, nous proposons une nouvelle architecture décisionnelle dédiée dans laquelle le processus ETL parallèle est décrit. Pour déployer notre approche, nous allons nous fixer sur la fonctionnalité *CDC*. Pour plus de clarté, nous présentons la fonctionnalité *CDC* à la fois dans un environnement classique et dans un environnement *Big Data*.

5.1. Fonctionnalité ETL vs Tâche ETL

Il faut noter qu'il y a une différence entre une fonctionnalité et une tâche d'ETL. La fonctionnalité d'ETL est une fonction de base qui prend en charge un bout de traitement dans l'ETL telles que *Changing Data Capture (CDC)*, *Data Quality Validation (DQV)*, *Surrogate Key (SK), Slowly Changing Dimension (SCD), Surrogate Key Pipeline (SKP)*. La tâche d'ETL, quant à elle, est une instance d'une fonctionnalité d'ETL. Par exemple, soit un processus ETL contenant deux tâches *SK1* et *SK2* qui permettent de générer respectivement une clé de substitution pour les tuples à insérer dans les tables de dimension *PRODUIT* et *CLIENT* d'un *DW*. *SK1* et *SK2* sont deux tâches différentes mais toutes les deux se basent sur *SK*. Dans ce qui suit, nous décrivons un processus ETL en fonction de ses fonctionnalités.

5.2. Principe de PF-ETL

Le processus *PF-ETL* que nous proposons est orienté fonctionnalités et est basé sur le paradigme *MR*. Pour chacune de ses fonctionnalités, nous appliquons le même principe que celui adopté, à un niveau processus, par l'approche « ETL basé sur le modèle *MR* » de la figure 3. Comme le montre la figure 4, le processus ETL s'exécute en une seule instance alors que chacune de ses fonctionnalités s'exécute en plusieurs instances.

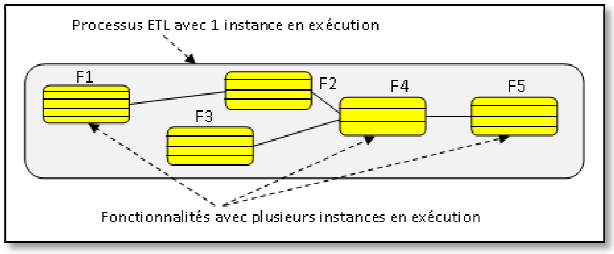


Figure 4. PF-ETL (Parallel Functionality-ETL)

Par exemple, la fonctionnalité F4 (forme ovale) qui s'exécute en trois instances (fragments séparés par des traits), a reçu des données en entrée à partir de F2 et F3. Ces inputs sont partitionnés et chacune des trois partitions est soumise à une instance de F4 (mapper). Les résultats partiels fournis par les trois mappers sont fusionnés par des reducers pour constituer le résultat de F4. *PF-ETL* n'est pas suffisant pour obtenir de bonnes performances à un niveau processus si celui-ci présente beaucoup de fonctionnalités séquentielles. Le concepteur pourra, après avoir constitué un processus selon *PF-ETL*, le paramétrer pour l'exécuter, lui aussi, en plusieurs instances. Il s'agit d'une approche hybride qui adopte, en même temps, les principes de l'approche « ETL basé sur le modèle *MR* » et ceux de *PF-ETL* comme le montre la figure 5. Il faut noter que l'approche hybride exige plus de

ressources. Lorsque le processus ETL s'exécute selon l'approche *PF-ETL* et atteint *F4*, il nécessitera dix tâches si *F4* s'exécute en dix instances. Le même processus exécuté selon l'approche hybride, nécessitera cent tâches parallèles si, en plus des dix instances de *F4*, lui-même s'exécute en dix instances.

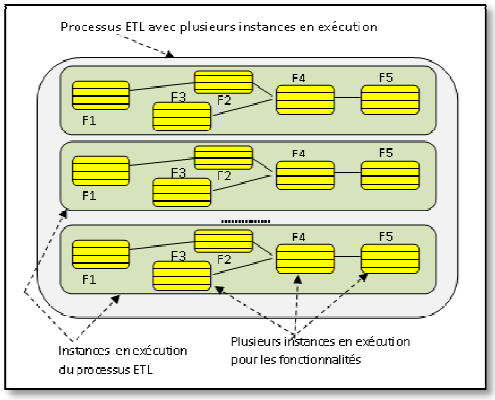


Figure 5 : PF-ETL Hybride

5.3. Changing Data Capture (CDC) dans un environnement classique

La plupart des travaux qui existent dans la littérature (cf. Liu et *al.*, 2011), performent uniquement la phase (T) de l'ETL. Or, la phase (E) est problématique lorsqu'il s'agit de données massives. La fonctionnalité *CDC* consiste à identifier, dans les systèmes sources, les données affectées par des changements *(INSERT, UPDATE, DELETE*) pour être capturées et traitées dans le rafraichissement du *DW* (cf. Kimball et Caserta, 2004). La technique la plus utilisée est celle basée sur les *snapshots*. Comme le montre la figure 6, la table source notée *TS* contient des données dans leur version récente (*snapshot j*), la table du *DSA* notée *TSvp* contient les mêmes données dans leur version précédente (*snapshot j-1*). Le processus montre deux étapes (1) le chargement complet de *TS* dans le *DSA* et (2) la comparaison entre *TS* et *TSvp.* Les tuples *01, 03 et 23* seront rejetés par *CDC* puisqu'ils ne présentent aucun changement depuis le chargement précédent (*TSvp*). Par contre, *02* et *22* ont connu des mises à jour et seront capturés comme modification (*UPDATE*). Les tuples *25, 26* et *27* seront capturés comme nouveaux (*INSERT*) puisqu'ils n'apparaissent pas dans *TSvp*. Le tuple *24*, quant à lui, est considéré comme suppression (*DELETE*) puisqu'il apparait dans *TSvp* mais n'apparait plus dans *TS*.

L'algorithme 1, décrit la fonctionnalité *CDC* dans un schéma classique. La ligne 5 charge *TS* dans *DSA*. Les lignes *6* et *7* trient *TS* et *TSvp* selon la clef *#A1* afin de détecter les insertions *(TSvp.A1>TS.A1),* suppressions *(TSvp.A1<TS.A1)* et modifications *(TSvp.A1=TS.A1* avec une différence dans au moins un attribut*).* Les autres cas seront rejetés *(*copie similaire du tuple dans *TS* et *TSvp*) puisqu'aucun changement n'a eu lieu depuis le dernier rafraichissement. Pour détecter les cas de

modifications, nous appliquons une fonction de hachage connue sous le nom de *CRC* (*Cyclic Redundancy Check*) (Freivald et *al*., 1999) sur les tuples de TS et de TSvp ayant la même valeur de la clef.

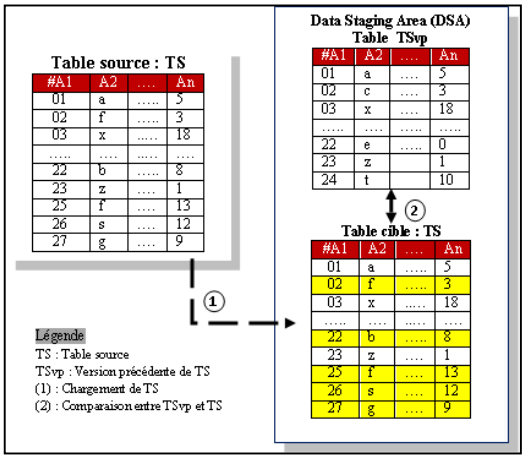


Figure 6 : Principe de la fonctionnalité CDC basée sur des snapshots

Algorithme 1. Fonctionnalité CDC

CDC_SNAPSHOTS
entrées: TS, TSvp
sortie: DSA
variables VT1 : enregistrement de type TSvp . VT2 : enregistrement de type TS
TS (DSA) ← TS (système source)
trier TSvp sur A1 par ordre croissant
trier TS sur A1 par ordre croissant
tantque non fin (TS) **et non fin** (TSvp) **faire**
    lire (TSvp, TV1)
    lire (TS, TV2)
    si VT1.A1= VT2.A1
        si CRC(VT1) # CRC(VT2) **alors**
            capturer le tuple VT2 comme modification (*update*)
        fin condition ;
    sinon
        si VT1.A1 < VT2.A1 **alors**
            capturer le tuple VT1 comme suppression (*delete*)
        sinon
            capturer le tuple VT2 comme insertion (*insert*)
        fin condition
    fin condition
fin boucle
tant que non fin (TS) **faire**
    capturer le tuple VT2 comme insertion (*insert*)

```
25:       lire (TS, VT2)
26:   fin boucle
27:   tant que non fin (TSvp) faire
28:       capturer le tuple VT1 comme suppression (delete)
29:       lire(TSvp, VT1)
30:   fin boucle
31:   sortie
```

5.4. Changing Data Capture (CDC) dans l'approche PF-ETL

Pour proposer un schéma de *CDC* dans un environnement *Big Data*, nous considérons que *TS* et *TSvp* sont massives, l'exécution de *CDC* se fera sur un *cluster* et nous adoptons le modèle *MR*. Le schéma classique de *CDC* sera complété par de nouveaux aspects à savoir (1) partitionnement des données, (2) utilisation de tables d'index (*Lookup)*, (3) processus de capture d'insertions et de modifications et (4) processus de capture de suppressions.

5.4.1. Partitionnement des données.

Nous adoptons la règle « *diviser pour régner* » qui consiste, dans ce contexte, à partitionner *TS* et *TSvp* avec une volumétrie excessive de manière à obtenir des volumes habituels de données. *TS* sera chargée dans le *DSA* puis partitionnée pour permettre un traitement parallèle des partitions générées. *TSvp* sera partitionnée pour éviter de rechercher les tuples de *TS* dans un grand volume de données.

5.4.2. Tables Lookup

Pour éviter de rechercher un tuple dans toutes les partitions de *TSvp* et de *TS*, nous utilisons des tables *Lookup* notées respectivement *LookupTSvp* et *LookupTS*. Elles permettent d'identifier, pour un tuple donné, la partition qui pourra le contenir. Voici quelques détails sur l'utilisation des tables *Lookup* :

- *LookupTSvp* et *LookupTS* contiennent, respectivement, les valeurs min et max des clefs naturelles (*#NK*) de chaque partition de *TSvp* et de *TS*;
- Pour un tuple T_i de *TS*, il s'agit de rechercher, dans *LookupTSvp,* la partition $Ptsvp_k$ de *TSvp* vérifiant l'expression :

$$LookupTSvp.NKmin \leq Ti.NK \leq LookupTSvp.NKmax \qquad (1)$$

- Il est possible qu'un tuple T_i de *TS,* vérifiant bien l'expression (1), n'existe pas dans $Ptsvp_k$; il s'agit, dans ce cas, d'un nouveau inséré dans *TS* ;
- Pour un tuple T_j de *TSvp*, il s'agit de rechercher, dans *LookupTS,* la partition Pts_k de *TS* vérifiant l'expression :

$$LookupTS.NKmin \leq Tj.NK \leq LookupTS.NKmax \qquad (2)$$

– Il est possible qu'un tuple T_j de *TSvp,* vérifiant bien l'expression (2), n'existe pas dans Pts_k ; il s'agit, dans ce cas, d'un tuple supprimé dans *TS*.

5.4.3. Processus IUDCP et DDCP

Nous proposons deux processus parallèles dans le nouveau schéma de *CDC* qui prennent en charge (1) la capture des insertions et modifications (*IUDCP*) et (2) la capture des suppressions (*DDCP*). Chacun s'exécute en plusieurs instances parallèles. Chaque instance d'*IUDCP* et de *DDCP* prennent en charge respectivement une partition de *TS* et une partition de *TSvp*. Si *TS* et *TSvp* sont partitionnées chacune, en 10 partitions, *CDC* s'exécutera en 20 tâches parallèles.

5.4.3.1. *IUDC Process* : Traitement parallèle des partitions *TS*

La figure 7 décrit *IUDCP*. Chaque partition Pts_i est confiée à Map_i chargé de vérifier, pour chacun de ses tuples, l'existence de celui-ci dans *TSvp*. Pour ce faire, le *mapper* passe par *LookupTSvp* pour identifier la partition $Ptsvp_k$ qui pourra le contenir (*#NK*). Dès que la partition $Ptsvp_k$ est identifiée, trois cas peuvent se présenter : (1) *#NK* inexistante dans $Ptsvp_k$; il s'agit d'une insertion (*INSERT*), (2) *#NK* existe avec une copie similaire du tuple dans $Ptsvp_k$; le tuple est rejeté (3) *#NK* existe dans $Ptsvp_k$ avec un changement ; il s'agit d'une modification *(UPDATE).*

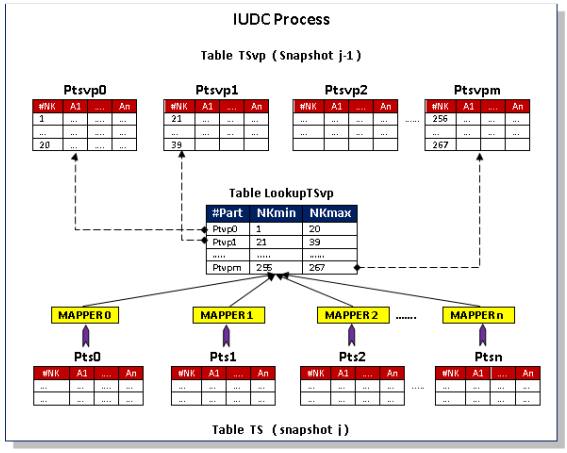


Figure 7 : Principe de fonctionnement du processus IUDCP

5.4.3.2. *DDC Process* : Traitement parallèle des partitions de *TSvp*

Dans *IUDCP,* deux *mappers* traitant deux partitions différentes Pts_i et Pts_j peuvent être orientés par *LookupTSvp* vers une même partition $Ptsvp_k$. Ainsi, un

même tuple de celle-ci pourra alors être capturé plusieurs fois comme suppression. C'est la raison pour laquelle, nous avons proposé un autre processus appelé *DDCP*.

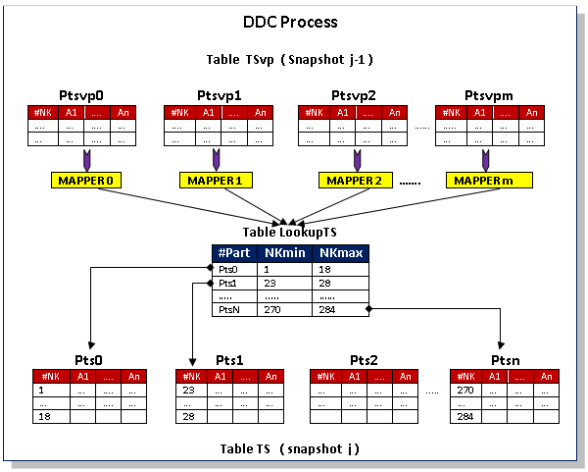


Figure 8 : Principe de fonctionnement du processus DDCP

Comme le montre la figure 8, chaque partition $Ptsvp_i$ est confiée à Map_i chargé de vérifier, pour chacun de ses tuples, l'existence de celui-ci dans *TS*. Pour ce faire, le mapper passe par *LookupTS* pour identifier Pts_k de *TS* qui pourra contenir le tuple et dès que celle-ci est identifiée, nous retenons le cas où le tuple n'existe pas *(DELETE)*.

5.4.4. Algorithmes

Le programme principal *CDC_BigData* consiste à partitionner *TS* et *TSvp*, générer *LookupTS* et *LookupTSvp* et enfin lancer, en parallèle, *IUDCP et DDCP*. L'algorithme 2 montre comment *IUDCP* pousse les partitions de *TS* vers le processus *MR* qui s'exécutera en un nombre d'instances égal au nombre de partitions de *TS*. L'algorithme décrivant *DDCP* fonctionne avec le même principe.

Algorithme 2. Processus IUDCP

```
1:  IUDCP
2:  entrées: LookupTS
3:  variables VT: enregistrement de type LookupTS ;  i: integer ;  i←1
4:  ouvrir (LookupTS)
5:  tant que non fin (LookupTS) faire
6:        lire (LookupTS, VT)
7:        soumettre VT.Part à iu_mapreader(i); i←i+1
8:  fin boucle
9:  iu_map()     // appel de la fonction iu_map()
```

L'algorithme 3 est chargé de capturer les insertions et les modifications effectuées dans *TS*. Une partition Pts_i sera traitée par une instance de *iu_map()*. Les lignes 6-12 recherchent $Ptsvp_K$ qui pourra contenir le tuple lu dans la ligne 7. Les lignes 13-18 traitent le cas où $Ptsvp_K$ est localisée. Celle-ci est parcourue jusqu'à détection du tuple. Les lignes 19-22 capturent les modifications en identifiant les tuples semblables (même valeur de la clef*)* de *TS* et *TSvp* et vérifient les éventuels changements avec la fonction CRC (ligne 20) ayant affecté le tuple, auquel cas le tuple est capturé comme modification (*UPDATE*). Les lignes 23-24 traitent le cas où le tuple n'existe pas dans la partition, celui-ci est capturé comme insertion (*INSERT*). Les lignes 26-27 capturent comme insertion (*INSERT*) les tuples où *LookupTSvp* montre qu'ils n'apparaissent dans aucune des partitions de *TSvp.*

Algorithme 3. Tâche de capture d'insertions et modification de données

```
1:  IU_MAP(Pts)
2:  entrées: Ptsvp, LookupTS
3:  sortie: DSA
4:  VT1 : enreg. de type TS, VT2 : enreg. de type LookupTSvp ; VT3 : enreg. de type  Ptsvp
5:  ouvrir (Pts)
6:  tantque non fin (Pts) faire
7:      lire (Pts, VT1)
8:      ouvrir (LookupTSvp)
9:      lire (LookupTSvp, VT2)
10:     tant que non fin (LookupTSvp)  et VT1.NK > VT2.NKmax faire
11:             lire (LookupTSvp, VT2)
12:     fin boucle
13:     si non fin (LookupTSvp) alors
14:             ouvrir (VT2.Part)
15:             lire (VT2.Part, VT3)
16:             tant que non fin (VT2.Part) et VT1.NK > VT3.NK
17:                     lire (VT2.Part, VT3)
18:             fin boucle
19:             si VT1.NK=VT3.NK alors
20:                     si CRC(VT1) # CRC(VT3) alors
21:                             capturer le tuple VT1 comme modification (update)
22:                     fin condition
23:             sinon
24:                     capturer le tuple VT1 comme insertion (insert)
25:             fin condition
26:     sinon
27:             capturer le tuple VT1 comme insertion (insert)
28:     fin condition
29: fin boucle
30: sortie
```

L'algorithme 4 est chargé de capturer les suppressions. Une partition $Ptsvp_i$ sera traitée par une instance de *d_map()*. Les lignes 6-12 localisent la partition Pts_i qui pourra contenir le tuple lu dans la ligne 7. Les lignes 13-18 traitent le cas où la

partition $Ptsvp_k$ est localisée. Si celle-ci ne contient pas le tuple (ligne 19), il sera capturé comme suppression (ligne 20). Les lignes 22-24 capturent comme suppression les tuples où *LookupTS* montre qu'ils n'apparaissent dans aucune partition de *TS*.

Algorithme 4. Tâches de capture de suppression de données

```
1:  D_MAP(Pts)
2:  entrées: Pts, LookupTSvp
3:  sortie: DSA
4:  VT1 : enreg de type TSvp ; VT2 : enreg de type LookupTS ; VT3 : enregide type Pts
5:  ouvrir (Ptsvp)
6:  tantque non fin (Ptsvp) faire
7:      lire (Ptsvp, VT1)
8:      ouvrir (LookupTS)
9:      lire (LookupTS, VT2)
10:     tant que non fin (LookupTS) et VT1.NK > VT2.NKmax faire
11:         lire (LookupTS, VT2)
12:     fin boucle
13:     si non fin (LookupTS) alors
14:         ouvrir (VT2.Part)
15:         lire (VT2.Part, VT3)
16:         tant que non fin (VT2.Part) et VT1.NK > VT3.NK
17:             lire (VT2.Part, VT3)
18:         fin boucle
19:         si VT1.NK < VT3.NK alors
20:             capturer le tuple VT1 comme suppression (delete)
21:         fin condition
22:     sinon
23:         capturer le tuple VT1 comme suppression (delete)
24:     fin condition
25: fin boucle
26: sortie
```

6. Implémentation et expérimentations

Notre prototype *PF-ETL* a été développé avec *java* 1.7 (*java* ™ *SE Runtime Environment*) sous *Netbeans IDE* 7.4. Nous avons déployé un *cluster* avec dix nœuds dont chacun possède un processeur *Intel Core* i5-2500 *CPU* @ 3.30 *GHz* x 4, *OS type* : 64-bit, *Ubuntu* 12.10 et le framework *Hadoop* 1.2.0.

Pour valider notre approche *PF-ETL*, nous avons défini deux scénarios d'expérimentations. Notre objectif est de comparer *PF-ETL* basé sur les fonctionnalités (niveau de granularité fin) avec un ETL parallèle à un niveau processus (niveau de granularité élevé). Nous présentons dans ce qui suit les résultats obtenus selon le deuxième scénario. Les tests selon le premier scénario étant en cours d'élaboration. Nous espérons avoir les premiers résultats très prochainement afin de procéder à une étude comparative entre les deux scénarios et

montré que la décomposition d'un processus ETL en un ensemble de fonctionnalités permet le passage à l'échelle plus facilement notamment dans le cadre des *Big Data*. Nous avons mené des expérimentations qui nous permettent de mesurer les performances d'un processus ETL constitué de huit fonctionnalités (1) *Extraction* des données (2) *partitionnement* et *distribution* des données sur le *cluster* (3) *projection* (4) *restriction* avec *NOT NULL* (5) *extraction* de l'année avec *YEAR()* (6) *agrégation* avec *COUNT()* (7) *fusion* des données (8) *chargement* dans un *DW*. Les temps d'exécution sont évalués en faisant varier la taille des données sources et le nombre de tâches *MapReduce* s'exécutant en parallèle. Le tableau 2 montre que la distribution des données et des traitements sur un nombre de tâches MapReduce plus important donne plus de capacité à l'ETL pour faire face aux données massives et améliore les temps d'exécution.

Tableau 2. Temps d'exécution (mn) de l'ETL à un niveau processus

	Nombre de tâches			
Taille (GO)	5	7	8	11
30	39,37	34,2	30,6	26,15
50	53	51	48,5	45,2
80	79	73,4	68	62,8
100	102	93,7	85,4	80,3

7. Conclusion

Les SID se trouvent aujourd'hui face à un défi majeur caractérisé par les *Big Data* et doivent donc s'adapter aux nouveaux paradigmes tels que le *cloud computing* et le modèle *MR*. Le processus ETL étant le cœur d'un SID puisque toutes les données destinées à l'analyse y transitent. Il faudra étudier de manière profonde l'impact de ces nouvelles technologies sur l'architecture, la modélisation et les performances de l'ETL. Dans ce contexte, nous avons proposé une approche parallèle pour les processus ETL dont les fonctionnalités s'exécutent selon le modèle *MR*. Dans un futur proche, nous envisageons de mener des tests à plus grande échelle. Par ailleurs, le *cloud computing* est un environnement qui dispose de toutes les ressources en termes d'infrastructures, de plateformes, en particulier des plateformes *MR* et d'applications pour offrir des services de qualité avec des coûts raisonnables. Notre approche *PF-ETL* dispose des qualités techniques en vue d'une migration vers un environnement *cloud*. Dans cette perspective, nos travaux futurs devront prendre en considération des aspects liés à la virtualisation et à l'architecture *SOA*.

Références

Arbab F. (2004). Reo: A Channel-based Coordination Model for Component Composition, *Mathematical Structures in Computer Science archive, Volume 14, Issue 3, p. 329–366.*

Bala M. et Alimazighi Z. (2013). Modélisation de processus ETL dans un modèle MapReduce, *Conférence Maghrébine sur les Avancées des Systèmes Décisionnels (ASD'13), p. 1–12,* Marrakech, Maroc.

Dean J. et Ghemawat S. (2004). MapReduce: Simplified Data Processing on Large Clusters, *6th Symposium on Operating Systems Design & Implementation (OSDI '04), p. 137–150,* San Francisco, CA, USA.

El Akkaoui Z. et Zemányi E. (2009). Defining ETLWorfklows using BPMN and BPEL, *DOLAP'09, p. 41–48*, Hong Kong, China.

El Akkaoui Z., Zimányi E., et Mazón J. N., Trujillo J. (2011). A Model-Driven Framework for ETL Process Development, *DOLAP'11, p. 45–52,* Glasgow, Scotland, UK.

Freivald, M. P., Noble A. C. et Richards M. S. (1999). Change-detection tool indicating degree and location of change of internet documents by comparison of cyclic-redundancy-check (crc) signatures. *US Patent 5,898,836.*

Ghemawat S. et Dean J. (2008). MapReduce: Simplified Data Processing on Large Clusters, *Communications of the ACM*, *Volume 51, Issue 1, p. 107-113,* New York, USA.

Han, J., E. Haihong, G. Le, et J. Du (2011). Survey on NoSQL database. *6th International Conference on Pervasive Computing and Applications (ICPCA'11), pp. 363–366*. Port Elizabeth, South Africa.

Kimball R. et Caserta J. (2004). *The Data Warehouse ETL Toolkit, p. 105–254,* Wiley Publishing, Inc., Indianapolis, USA.

Liu X., Thomsen C., et Pedersen T. B. (2011). ETLMR : A Highly Scalable Dimensional ETL Framework based on Mapreduce, *in proceedings of 13th International Conference on Data Warehousing and Knowledge, p. 96–111*, Toulouse, France.

Misra S., Saha S.K., et Mazumdar C. (2013). Performance Comparison of Hadoop Based Tools with Commercial ETL Tools – A Case Study, *Big Data Analytics (BDA'13), LNCS 8302, p. 176–184, 2013,* Mysore, India.

Mohanty S., Jagadeesh M., et Srivatsa H. (2013). *Big Data Imperatives*, *p. 1–22,* Apress, NY, USA.

Oliveira B. et Belo O. (2013). Using Reo on ETL Conceptual Modelling A First Approach, *DOLAP'13, p. 55–60*, San Francisco, CA, USA.

Sosinsky B. (2011). *Cloud Computing Bible*, *p. 45–88,* Wiley Publishing, Inc, Indiana, USA.

Thomsen C. et Pedersen T. (2009a). Pygrametl: A Powerful Programming Framework for Extract-Transform-Load Programmers. *In Proc. of DOLAP, p. 49–56,* Hong Kong, China.

Thomsen C. et Pedersen T. (2009b). Building aWebWarehouse for Accessibility Data, *In Proc. of DOLAP'09, p. 43–50*, Hong Kong, China.

Trujillo J. et Luján-Mora S. (2003). A UML Based Approach for Modeling ETL Processes in Data Warehouses*, ER 2003, LNCS 2813, p. 307–320, Springer-Verlag* Berlin Heidelberg.

Vassiliadis P., Simitsis A., et Skiadopoulos S. (2002). Conceptual Modeling for ETL Processes, *DOLAP'02, p. 14–21,* McLean, Virginia, USA.

Petits textes pour grandes masses de données

Cyril Labbé — Damien Bras — Claudia Roncancio

Univ. Grenoble Alpes, LIG, F-38000 Grenoble, France
first.last@imag.fr

RÉSUMÉ. *Maîtrisée, l'omniprésence des données offre aujourd'hui un potentiel de services sans précédent. Dans une optique centrée personne, nous proposons une solution étendue pour l'exploitation de masses de données en flux. Notre solution, nommée* Stream2Text*, s'appuie sur un raffinement personnalisé et continu des données, et produit des textes (en langue naturelle) qui résument de manière personnalisée les données intéressantes pour l'utilisateur. Les flux textuels produits permettent un monitoring adapté à un large spectre d'utilisateurs et peut être partagé dans un réseau social ou utilisé individuellement à partir de dispositifs mobiles.*

ABSTRACT. *When controlled, omnipresence of data can leverage a potential of services never reached before. We propose an user driven approach to take advantage of massive data streams. Our solution named* Stream2Text *rests on a personalized and continual refinement of data to generates texts (in natural language) that give in a tailored synthesis of relevant data. This textual stream enables monitoring by a wide range of users. It can also be shared on social networks or be used individually on mobile devices.*

MOTS-CLÉS : *flux données, flux de textes, résumé de données, préférences, génération de textes.*

KEYWORDS: *data stream, texts stream, data summarization, preferences, texts generation.*

1. Introduction

Avec l'omniprésence des données, un grand nombre d'informations sont accessibles partout, tout le temps au travers de divers supports disposant ou non d'écrans de tailles variables. Les données sont complexes, nombreuses, changeantes, incertaines, caractéristiques connues comme les 4V du *Big Data, Volume, Variety, Velocity, Veracity*. Face à cette masse de données, nous nous plaçons dans une optique centrée personne : l'information extraite nécessite une adaptation du contenu et de la forme pour être assimilable par l'utilisateur. L'adaptation du contenu, permettant la maîtrise du volume de données, nécessite de prendre en compte les préférences courantes de l'utilisateur et les différents médium disponibles.

Le monitoring continu de flux de données a comme caractéristique l'intégration de données produites à la volée et de données persistantes qui les enrichissent. L'objectif de notre travail est de faciliter l'exploitation des données complexes en facilitant leur résumé. Pour cela, nous montrons la possibilité de générer des flux textuels personnalisés qui résument en langue naturelle des flux de données. Un tel flux textuel permet un monitoring adapté à un large spectre d'utilisateurs et peut être partagé dans un réseau social ou utilisé individuellement à partir de dispositifs mobiles.A titre d'exemple, considérons le domaine financier où des utilisateurs s'intéressent aux performances des marchés. On considérera des données telles que le taux de volatilité des actions, la situation économique du pays émetteur des actions, des ordres d'achat et de vente d'actions. Dans ce contexte, le volume global des données rend difficile (ou inintéressant) la récupération totale des données par l'utilisateur. A une requête du type *Connaître les opérations du jour*, l'utilisateur préférera une requête plus centrée sur ses intérêts :

Avoir un compte rendu journalier des opérations
de la journée les plus « intéressantes » pour moi.

Ainsi, l'objectif de nos recherches est de produire un compte rendu textuel personnalisée qui décrit les informations disponibles grâce aux préférences courantes de l'utilisateur. Dans le cadre de ce travail, nous intégrons des préférences qualitatives contextuelles qui donnent au système des connaissances sur les priorités de l'utilisateur. Les préférences reflètent des informations telles que

Pour les actions issues des pays en situation économique difficile,
je préfère les actions avec une faible volatilité ces trois derniers jours.

Le système personnalisera les comptes rendus textuels (dits "résumés" par la suite) produits à la volée pour refléter les informations prioritaires pour l'utilisateur. Le traitement continu de flux de données et la connaissance des concepts du domaine permettront de produire des flux de textes courts répondant à des demandes telles que :

Je souhaite un résumé toutes les 2 heures
des 50 dernières opérations sur mes actions préférées.

Les résumés personnalisés produits pourront être lus ou écoutés (voix de synthèse), notamment en situation de mobilité (voiture par exemple). Ce type de compte rendu facilite également l'accès aux informations par des personnes en situation de handicap.

Dans la suite de l'article, nous présenterons les éléments essentiels de notre proposition, nommée **Stream2text**. Les aspects liés à l'évaluation de requêtes continues

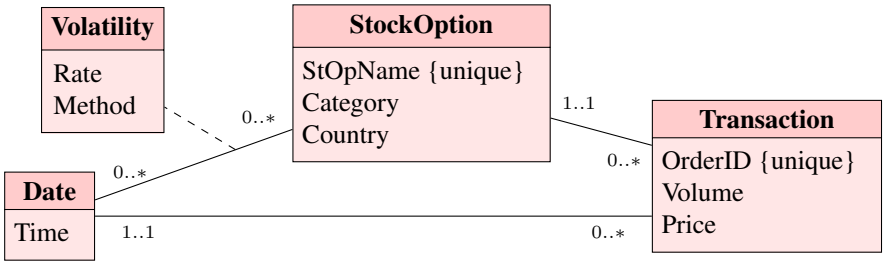


Figure 1. *Modèle des données pour l'exemple du marché boursier*

avec préférences sont introduits et nous développerons de manière plus complète la génération des textes. A notre connaissance, cet effort est le premier à mettre en œuvre le continuum permettant la production automatique de flux de textes personnalisés qui résument des flux de données. Ces résumés textuels offrent la possibilité d'accéder à des informations difficilement exploitables par beaucoup d'utilisateurs.

L'article est organisé ainsi : la section 2 précise l'exemple, la 3 donne une vue globale de **Stream2text**. La section 4 présente des outils et fondements théoriques pour interroger, personnaliser et agréger des données. La section 5 définit les opérateurs de *textualisation* et la 6 expose les choix d'implémentation et les expérimentations réalisées. Les sections 7 et 8 présentent les travaux connexes et notre conclusion.

2. Motivation et cas d'étude

La suite de cet article traite, dans un modèle unifié, des données ayant des caractéristiques de volatilité très différentes : les flux de données et les données persistantes. De manière générale, un flux de données est un ensemble potentiellement infini de n-uplets conformes à un schéma commun possédant un *timestamp*. Les données persistantes sont représentées sous forme de relations, *i.e.* ensemble fini de n-uplets conforme à un schéma commun.

En considérant le cas d'étude présenté en introduction, observons Luc, un investisseur qui suit les évolutions financières pour prendre ses décisions d'achats ou de ventes d'actions. Il dispose d'un accès à de nombreuses sources d'information temps réel sur l'état du marché (cf. schéma conceptuel en figure 1). Ces données sont pour partie des flux et pour partie des relations persistantes :

Relation *StockOption(StOpName, Category, Country)*. Cette relation est un catalogue d'actions incluant le nom des actions, leur catégorie et le pays où l'entreprise a son siège social. Les catégories sont, par exemple, 'Technologie' (IT) , 'Commodities' (Co) ou 'Manufacturing'.
Flux *Transaction(OrderID, TTime, StOpName, Volume, Price)*. Ce flux de données décrit les transactions boursières : l'heure ($TTime$), le nom de l'action ($StOpName$), le nombre de parts vendues ($Volume$) ainsi que le prix unitaire ($Price$).
Flux *Volatility(StOpName, ETime, Rate, Method)* Ce flux de données donne la volatilité ($Rate$) pour les actions (ampleur des variations du cours). Elle est calculée à une certaine date ($ETime$) avec une méthode ($Method$).

Afin de faciliter les prises de décisions, Luc doit pouvoir accéder à ces informations à toute heure et sur différents supports (fixe, mobile, avec ou sans écran, etc). De manière à réduire et mieux cibler le flux d'information selon des critères personnels du moment, il exprime des préférences qui doivent être prises en compte. Informellement, ses préférences combinent des informations qualitatives et quantitatives :

[P1] Pour les actions de la catégorie 'Co', Luc préfère celles qui ont une volatilité inférieure à 0.25. Par contre, pour la catégorie 'IT', Luc préfère les actions dont la volatilité est supérieure à 0.35.

[P2] Pour les actions dont la volatilité est actuellement plus grande que 0.35, Luc préfère les actions brésiliennes à celles du Venezuela.

[P3] Pour les actions dont la volatilité est actuellement plus grande que 0.35, Luc est intéressé par les transactions des 3 derniers jours et préfère celles dont le volume est supérieur à 1000 parts.

Les préférences de Luc peuvent être exprimées au moyen de règles de la forme
`SI` *un contexte est vérifié* `ALORS` *Luc préfère quelque chose*.

Ainsi, pour **[P1]** le contexte est : $StockOption.Category =' Co'$

et la préférence est : $Volatility.Rate \leq 0.25$ plutôt que $Volatility.Rate > 0.25$.

Ces règles de préférences peuvent impliquer des flux de données aussi bien que des données persistantes. Les demandes de compte rendu de Luc s'expriment dans cet environnement et peuvent être de nature continue :

[Q1] Chaque jour, un compte rendu des deux derniers jours pour l'action $Total$.

[Q2] Toutes les heures, donner le compte rendu de la dernière heure pour la catégorie 'IT' parmi les 100 transactions qui satisfont le mieux mes préférences.

[Q3] Toutes les heures, donner le compte rendu de la dernière heure des 100 transactions préférées de la catégorie 'IT'.

[Q4] Donner un compte rendu des 1000 dernières transactions concernant des titres français ayant une forte volatilité ($> 0,8$) et dont au moins une transaction a concerné un volume important (> 100).

L'extraction des données peut être faite de manière très précise selon les souhaits du moment. Ainsi, [Q2] travaille sur les 100 transactions préférées de Luc et en extrait celles de la catégorie IT. [Q3] va travailler sur les transactions de la catégorie IT et en extraire les 100 préférées. La fréquence du compte rendu est indiqué de manière temporelles (pour Q1, Q2 et Q3) ou positionnelle (pour Q4, toutes les 1000 transactions).

3. Architecture du système Stream2text

Cette section présente les grandes lignes du processus et l'architecture globale du système proposé.

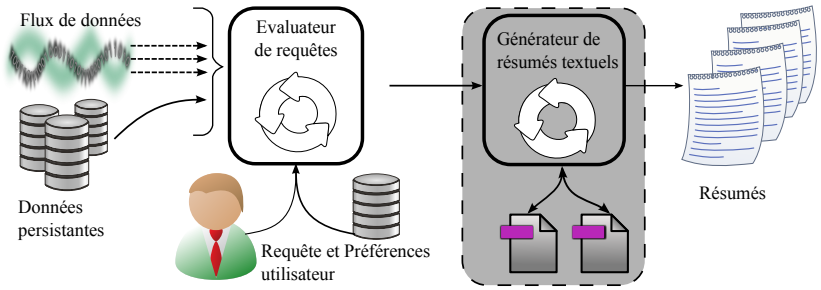


Figure 2. *Architecture globale de Stream2text*

Vue Globale du système : Stream2text (cf. figure 2) prend en entrée une requête utilisateur pour sélectionner les données pertinentes à résumer. Stream2text fournit un flux de textes résumant les données préférées de l'utilisateur.

La demande de l'utilisateur comporte d'une part "le point de vue" choisi par l'utilisateur et d'autre part une description de la fréquence et de la portée souhaitées pour les comptes rendus. Le point de vue permet d'orienter la rédaction du compte rendu en sélectionnant les données pertinentes. La fréquence rythme le déclenchement du processus de rédaction du compte rendu et la portée permet de limiter les données provenant des flux à un ensemble fini de données.

La requête de l'utilisateur s'exprime sur des flux et des données persistantes. Ceci est possible grâce à l'utilisation d'un modèle permettant d'exprimer de manière formelle les données qui sont nécessaires à la rédaction des comptes rendus. L'expression de préférences personnelles permet à l'utilisateur de limiter l'ensemble de données retournées au sous-ensemble satisfaisant le mieux ses choix personnels.

Une fois ces données disponibles, l'ensemble des données est agrégé de manière à obtenir un résumé numérique. Le résumé numérique est ensuite transcrit en langage naturel pour fournir un compte rendu textuel.

Architecture du générateur de textes : La rédaction du compte rendu textuel (cf. figure 3) s'appuie sur des informations concernant, d'une part le schéma des données et d'autre part les fonctions d'agrégations utilisées pour la phase de construction du résumé numérique.

Les informations nécessaires à la rédaction relatives au schéma sont principalement la "description" au format textuel des propriétés du schéma. Par exemple, le fait que *StOpName*=v peut être désigné dans un texte par "*L'action v*". Ou encore, qu'une instance de *StockOption* (représentée dans les données par un tuple $t \in StockOption$) peut se décrire par une phrase de type "*L'action* $t.StOpName$ *appartient à la catégorie* $t.Category$ *et l'entreprise est domicilié en* $t.Country$". Ces informations sont généralement disponibles puisqu'elles sont le produit des phases amonts (spécification) de la conception d'une application.

Les informations nécessaires à la rédaction du texte concernant les fonctions d'agrégation sont principalement la "description" au format textuel du sens associé aux

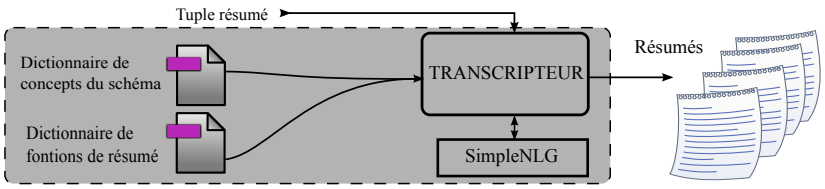


Figure 3. *Architecture du générateur de textes*

fonctions d'agrégation. Par exemple, une fonction d'agrégation $Avg(A)$ calculant la moyenne d'un attribut A peut s'exprimer comme "*La moyenne de* A *est* $Avg(A)$" ou encore "*Le* A *moyen(ne) est* $Avg(A)$".

4. Fondement théorique pour l'évaluation de requêtes

Cette section introduit les fondements pour l'évaluation de requêtes continues avec des préférences contextuelles ("évaluateur de requêtes" de la figure 2). Le lecteur familier avec ces aspects peut lire directement la section suivante.

4.1. *L'algèbre de flux ASTRAL*

Pour illustrer le propos, reprenons des requêtes de la section 2 sans les préférences ni le résumé textuel :
[Q1'] Chaque jour, informations concernant l'action $Total$ sur les 2 derniers jours.
[Q3'] Toutes les heures, informations concernant la catégorie 'IT' .
[Q4'] Informations pour les 1000 dernières transactions des actions françaises ayant une forte volatilité et dont au moins une transaction a un volume important.

Ces requêtes peuvent être exprimées en utilisant l'algèbre ASTRAL (Petit *et al.*, 2012b). Cette algèbre formalise l'expression de requêtes continues ou instantanées impliquant conjointement des flux et des relations. Ci-dessous, nous donnons quelques définitions nécessaires à la compréhension de ces requêtes qui permettent de traiter les données à la volée.

ASTRAL différencie les concepts de flux et de relation temporelle (Arasu *et al.*, 2004) qui sont notés S et R dans la suite. Un flux S est un ensemble potentiellement infini de n-uplets s ayant un schéma commun contenant deux attributs particuliers : un *timestamp* t, et une position [1] dans le flux p. Une *relation temporelle* R est une fonction qui fait correspondre à un identifiant temporel t un ensemble de n-uplets $R(t)$ ayant un même schéma. Les opérateurs de l'algèbre relationnelle (sélection σ, projection π, jointure $\bowtie$) sont étendus aux relations temporelles et σ et π aux flux. Ainsi, $\sigma_{Volume>10}(Transaction)$ est le flux des transactions dont le $Volume$ dépasse 10.

Une relation temporelle peut être extraite d'un flux à l'aide d'un opérateur de fenêtre. ASTRAL permet d'exprimer de nombreux types de fenêtres (Petit *et al.*, 2010) dont les fenêtres positionnelles comme, par exemple, l'ensemble des n derniers n-uplets, tous les m n-uplets, et les fenêtres temporelles comme, par exemple, l'ensemble des n-uplets arrivés pendant les x dernières secondes toutes les y secondes, ou

1. La notion de *batch* (Petit *et al.*, 2012b) ne sera pas utilisée dans cet article.

encore des fenêtres, dites *cross domain*, qui utilisent des positions et des *timestamp*. Par exemple, les n derniers n-uplets arrivés toutes les y secondes. Dans la suite de l'article, les fenêtres les plus utiles sont les suivantes :

– $S[L]$ est une fenêtre qui contient le dernier n-uplet du flux S (L pour $Last$) ;

– $S[N$ slide $\Delta]$ est une fenêtre de taille N glissant de Δ chaque Δ. N et Δ peuvent être, au choix, une durée ou un nombre de n-uplets.

Un flux peut être généré à partir d'une relation temporelle en utilisant un opérateur *streamer*. Par exemple, $I_S(R)$ produit le flux des n-uplets insérés dans la relation R. La jointure entre deux flux ou entre un flux et une relation s'exprime à l'aide des opérateurs de fenêtre, de *streamer* et d'une jointure sur des relations temporelles. Dans la suite on notera :

$$S \bowtie_c R = I_S(S[L] \bowtie_c R).$$

Ce flux $S \bowtie_c R$ contient les n-uplets générés par le flux S et ceux générés par des mises à jour dans R. On définit aussi l'opérateur *semi-sensitive-join* (noté $\bowtie\!\!\!\!\blacktriangleleft$) qui produit un flux résultant de la jointure entre le dernier n-uplet d'un flux et une relation temporelle à la date d'émission du n-uplet :

$$S \bowtie\!\!\!\!\blacktriangleleft_c R = I_S(S[L] \bowtie_c R(\tau_S(S[L])))$$

où τ_S est la fonction qui retourne la date d'émission d'un n-uplet dans le flux S. A titre d'exemple, voici la formulation des requêtes pré-citées :

[Q1'] $((Volatility \bowtie Transaction) \bowtie\!\!\!\!\blacktriangleleft (\sigma_{StOpName='Total'} StockOption))[2day \text{ slide } 1day]$ (1)

[Q3'] $((Volatility \bowtie Transaction) \bowtie\!\!\!\!\blacktriangleleft (\sigma_{Category='IT'} StockOption))[1h \text{ slide } 1h]$ (2)

[Q4'] $((\sigma_{rate>0.8} Volatility \bowtie \sigma_{Volume>100} Transaction) \bowtie\!\!\!\!\blacktriangleleft (\sigma_{Country='FR'} StockOption))[1000n \text{ slide } 1n]$ (3)

4.2. *Modèle de préférences contextuelles*

Cette section présente les principaux concepts du formalisme logique employé pour *spécifier et raisonner* avec des préférences (présentation détaillée dans (de Amo et Pereira, 2010 ; Petit *et al.*, 2012a)). Intuitivement, une *règle de préférence contextuelle* (une cp-règle) permet de comparer deux n-uplets d'une relation R compatibles avec un contexte :

$$\varphi : u \rightarrow Q_1(X) \succ Q_2(X)[W]$$

où $X \subseteq Attr(R)$, $W \subseteq Attr(R)$ et $X \notin W$; $Q_i(X)$ (pour $i = 1, 2$) est un prédicat évaluable sur un n-uplet de R ; u est aussi un prédicat ne faisant intervenir ni X ni W (cf. exemple 1). Deux n-uplets sont comparables à l'aide d'une cp-règle, si ils ont les mêmes valeurs pour les attributs entre crochets dans la règle (attributs *ceteris paribus*).

Une *théorie de préférences contextuelles* (*cp-théorie*) sur R est un ensemble fini de cp-règles. Si la cp-theorie satisfait certaines conditions de consistance alors elle établit un ordre strict partiel sur l'ensemble des n-uplets. Cet ordre partiel sera utilisé pour distinguer les données préférées d'un utilisateur.

Exemple 1 Soit P1 et P2 les préférences de notre exemple de la section 2. Elles s'expriment par la cp-théorie suivante sur le schéma $T(StOpName, Category, Country, ETime, Rate, Method)$:
– $\varphi_1 : Category = \text{`}Commodities' \rightarrow (Rate < 0.25 \succ Rate \geq 0.25), [Method]$
– $\varphi_2 : Category = \text{`}IT' \rightarrow (Rate \geq 0.35 \succ Rate < 0.35), [Method]$
– $\varphi_3 : Rate > 0.35 \rightarrow (Country = Brazil \succ Country = \text{`}Venezuela')$

Les préférences utilisateur mentionnées en figure 2 sont des cp-théories comme celles de l'exemple 1. L'algèbre ASTRAL est étendue avec les opérateurs de préférence dans un contexte dynamique de *flux de données*. La sémantique considérée pour la relation de préférence est celle *avec contrainte*. La motivation est que dans un scénario de flux de données, il est raisonnable d'imaginer que la notion de préférence a le même caractère dynamique que les données sur lesquelles elle est appliquée.

4.3. *Opérateurs de préférences pour ASTRAL*

Cette section présente l'approche d'intégration des préférences contextuelles dans l'algèbre ASTRAL. Il s'agit d'opérateurs algébriques qui peuvent être utilisés aussi bien avec des requêtes instantanées que continues et portant sur des données persistantes ou sur des flux.

Les opérateurs de préférences calculent les données préférées par rapport à une cp-théorie de référence. Chaque utilisateur donne au système ses préférences sous forme d'une cp-théorie qui constitue une sorte de *profil utilisateur*. Ces préférences sont utilisées si la personnalisation de requêtes est demandée. Concrètement, cette solution permet d'introduire des requêtes « top-k » par l'intégration de l'opérateur ***KBest*** qui sélectionne le sous-ensemble des k données préférées en accord avec la hiérarchie des préférences spécifiée par la cp-théorie.

Pour illustrer l'extension de l'algèbre ASTRAL avec l'opérateur de préférence ***KBest***, considérons la requête **Q2** de la section 2. Son expression avec l'opérateur de préférence est :

$$(\sigma_{Category=\text{`}IT'}(\mathbf{KBest_{100}}((Volatility \bowtie Transaction) \bowtie StockOption)))[1h \text{ slide } 1h] \quad (4)$$

Alors que **Q3** s'écrit : $KBest_{100}(Q2')$.

4.4. *Fonctions d'agrégations et résumé de données*

Pour établir les résumés textuels qui seront générés, Stream2text passe par la création d'un résumé structuré. Celui-ci est créé grâce aux fonctions d'agrégation de l'éva-

luateur de requêtes. Les fonctions utilisées dépendent du domaine d'application. Intuitivement, une fonction d'agrégation f est une fonction qui associe à un ensemble de n-uplets un unique n-uplet dont les attributs et les valeurs sont déterminés par f.

Dans la suite de l'article nous utiliserons la définition 1. Dans cette version, chaque attribut agrégé est renommé à l'aide de la fonction utilisée pour calculer la valeur agrégée :

Definition 1 (Opérateur d'agrégat) *Soit R une relation temporelle de schéma $A = \{a_i\}_{i=1..n}$, n attributs de R, Soit $f^j(\{A_i\}_{i\in\{1,n\}})_{j=1..m}$, m fonctions d'agrégations. L'opérateur d'agrégation $\mathcal{G}_{f^1,f^2,...f^m}$ agrège l'ensemble des n-uplets de R en un n-uplet à l'aide des fonctions f^j.*

$$\mathcal{G}_{f^1,f^2,...f^m}(R) = \{\cup_{j=1}^{m}(f^j, f^j(\{A_i\}_{i\in\{1,n\}}))\}$$

Rappelons que l'évaluateur de requêtes produit un ensemble de données qui sont ensuite agrégés dans un resumé structuré. Celui-ci est l'entrée du générateur de texte. Par exemple pour [Q1], notre utilisateur Luc peut demander un résumé avec la moyenne et la médiane pour le volume des transactions et les prix. Les valeurs obtenues par les fonctions d'agrégation constitueront le resumé structuré qui sera ensuite rédigé de manière appropriée en langue naturelle.

5. Opérateur de génération de textes

Pour automatiser la génération de textes, nous définissons des fonctions et des opérateurs permettant d'associer un texte à des données. Les sections 5.1 et 5.2 tirent profit des connaissances du schéma et des résumés numériques. La section 5.3 définit un opérateur de transcription qui transcrit une relation temporelle en langage naturel.

5.1. *Dictionnaire de concepts*

Ainsi, il est nécessaire d'associer à chaque propriété du modèle de données un fragment de texte. Ce dernier peut être utilisé pour nommer la propriété dans un texte. La définition 2 formalise cette notion sous la terminologie : *dictionnaire de concepts*.

Dans la suite, on considère une base de données relative à n_e entités/classes $\{E_i\}_{i=1..n_e}$. Chacune de ces classes ayant un ensemble n_i de propriétés/attributs $\{A_{i,j}\}_{j=1..n_i}^{i=1..n_e}$ dont certaines identifient de manière unique un objet dans la classe (attribut clef).

Definition 2 (Dictionnaire de concepts) *Un dictionnaire de concepts est une fonction $\mathcal{D}_c$ qui associe à chaque concept de la base de données (ie. propriété $A_{i,j}$) un groupe nominal GN. Ce groupe nominal peut être utilisé pour désigner le concept (la propriété $A_{i,j}$) dans un texte.*

$$\mathcal{D}_c(A_{i,j}) = \{GN\}_{i,j}$$

où $\{GN\}_{i,j}$ est un groupe nominal qui nomme le concept $A_{i,j}$ en langage naturel.

L'exemple 2 illustre quelques valeurs possibles de la fonction $\mathcal{D}_c$.

Exemple 2 (Exemples d'entrées du dictionnaire de concepts)

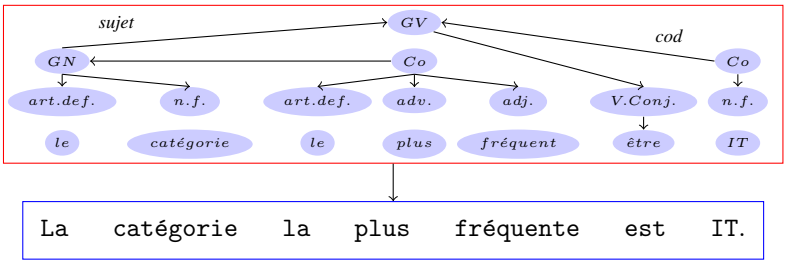
$$\mathcal{D}_c(StOpName) = \left\{ \begin{array}{cc} le & action \\ art.def. & n.f. \end{array} \right\} \quad \mathcal{D}_c(Price) = \left\{ \begin{array}{cc} le & prix \\ art.def. & n.m. \end{array} \right\}$$
$$\mathcal{D}_c(Price) = \left\{ \begin{array}{cc} le & cours \\ art.def. & n.m. \end{array} \right\}$$

Notons que la valeur de $\mathcal{D}_c$ pour un concept donné n'est pas unique. Ceci permet d'introduire diverses formulations et de limiter la répétition des textes.

5.2. *Dictionnaire de fonctions d'agrégat*

Le résumé textuel s'appuie sur un résumé numérique qui est obtenu à l'aide de fonctions d'agrégation. Pour la génération du texte à proprement parler, nous introduisons un dictionnaire de structures de phrase permettant d'exprimer le sens des fonctions d'agrégation et de la valeur calculée. Une structure de phrase est composée d'éléments de phrase ainsi que des relations entre ces éléments. Ces informations sont utilisées pour effectuer la *réalisation de surface* (cf. (Gatt et Reiter, 2009) et exemple 3) du texte c'est-à-dire son écriture en respectant les règles du langage naturel cible.

Exemple 3 (Structure de phrase et opération de réalisation) *Une structure de phrase, représentée sous forme de graphe, suivie de sa réalisation.*



La définition 3 propose une formalisation de la fonction permettant d'associer un texte au résultat d'une fonction d'agrégation F définie sur un ensemble de k attributs $\{a_i\}_{i=1..k}$. Le texte dépend à la fois des textes $\mathcal{D}_c(a_i)_{i=1..k}$ et du résultat du calcul de la fonction d'agrégation ie : $F(\{a_i\}_{i=1..k})$.

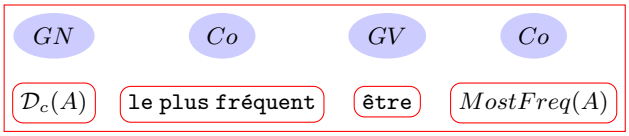
Definition 3 (Dictionnaire de fonctions d'agrégat) *Un dictionnaire de fonctions d'agrégat est une fonction $\mathcal{D}_f$ qui associe à une fonction d'agrégation $F(\{a_i\}_{i=1..k})$ une structure de phrase SP. La réalisation de SP permet de décrire le résultat de la fonction d'agrégation en langage naturel.*

$$\mathcal{D}_f(F) = \{\{\mathcal{D}_c(a_i)\}_{i=1..k}, GV, \{Co_i\}_{i=1..x}, F(\{a_i\}_{i=1..k}), \{R_j\}_{j=1..y}\}$$

où GV est un groupe verbal qui exprime la relation entre les attributs $\{a_i\}_{i=1..k}$ et $F(\{a_i\}_{i=1..k})$. $\{Co_i\}_{i=1..x}$ est un ensemble de x compléments (de nom, d'objet direct ou indirect) et $\{R_j\}_{j=1..y}$ un ensemble de y relations entre les éléments de la phrase.

L'exemple 4 est un exemple d'entrée du dictionnaire de fonctions d'agrégation.

Exemple 4 *Ci-après un exemple simplifié pour la fonction $MostFreq(A)$ qui* calcule la valeur la plus fréquente. *D'autres structures de phrases sont possibles.*



Une réalisation possible de cette structure de phrase est présentée dans l'exemple 3 avec l'attribut $Category$. La même fonction d'agrégat utilisée avec un autre attribut donnera une réalisation différente. Pour l'attribut $Country$, cela peut donner la réalisation suivante (en supposant $MostFreq(Country) = France$) :

Le pays le plus fréquent est France.

Ainsi, pour produire un texte, l'entrée du dictionnaire de fonction doit être *réalisée* et la fonction d'agrégat évaluée.

5.3. *Opérateur de transcription*

On dira qu'une relation temporelle (une fonction du temps) peut être *transcrite* s'il est possible de générer un ensemble de structures de phrases relatif à cette relation. Ainsi, transcrire en langage naturel la signification des données revient à produire un ensemble de structures de phrases décrivant les données d'une relation temporelle. Dans l'optique de la génération d'un résumé, la transcription intervient en fin de traitement lorsque le résumé structuré a été produit sous forme d'un n-uplet. La définition 4 explicite la forme des relations temporelles qui peuvent être transcrites.

Definition 4 (Relation transcriptible) *Une relation transcriptible est une relation temporelle R contenant un unique n-uplet t et : $\forall(A, v) \in t$ on a :*
– Soit A est un concept du dictionnaire (ie. $A \in Dom(\mathcal{D}_c)$)
– Soit $A = F$ où $F(\{a_i\}_{i=1..n})$ est une fonction d'agrégat utilisée pour agréger les valeurs des attributs $\{a_i\}_{i=1..n}$ en v. (ie. $F \in Dom(\mathcal{D}_f)$ et $(F, v) \in \mathcal{G}_F(R)$)

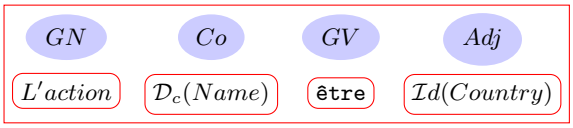
Nous définissons un opérateur de transcription qui retourne un ensemble de structures de phrases pour une relation transcriptible.

Definition 5 (Transcription de relation) *Un opérateur de transcription de relation $\mathcal{T}$ génère un ensemble de structures de phrases à partir d'une relation temporelle R de schéma F_i.*

$$\mathcal{T}(R) = \{\cup_i \mathcal{D}_f(F_i)\}$$

R étant une relation temporelle, $\mathcal{T}(R)$ est un ensemble de structures de phrases évoluant dans le temps. L'utilisation d'un opérateur de création de flux sur cet ensemble dépendant du temps permet d'insérer dans un flux les textes estampillés par la date de mise à jour de R.

De manière générale, l'apparition d'une fonction identité $\mathcal{I}d$ dans les attributs d'une relation transcriptible signale le "point de vue" adopté pour résumer les données. Le dictionnaire de fonctions peut avoir une entrée pour la fonction $\mathcal{I}d$ avec une phrase introductive du résumé (par exemple, « Résumé des informations... »).
REMARQUE. — [
Cas particulier des attributs clef] Il est possible qu'apparaisse dans la liste des attributs d'une relation transcriptible la fonction identité $\mathcal{I}d$ appliquée à la clef d'une entité. Cela signifie que la fonction de transcription doit décrire l'objet et non le "point de vue". Dans ce cas, la partie du n-uplet correspondant n'est pas un *résumé* mais un objet de la base. Il est donc nécessaire de disposer d'une entrée spécifique dans le dictionnaire de fonction pour $\mathcal{I}d$ sur les attributs clefs. Par exemple, $\mathcal{D}_f(\mathcal{G}_{\mathcal{I}d(Name),\mathcal{I}d(Country)})$ peut être défini comme suit :



Ainsi la réalisation de :

$$\mathcal{T}(\mathcal{G}_{\mathcal{I}d(Name),\mathcal{I}d(Country)}(\pi_{Name,Country}(\sigma_{Name='Total'}(StOpName)))) \quad (5)$$

est L' action Total est française.

L'opérateur de transcription peut ainsi être utilisé pour construire des textes à partir de toute requête Astral avec ou sans préférences utilisateur. Il faut noter que le texte généré dépend uniquement du contenu des données et des fonctions d'agrégat, la requête n'est qu'indirectement transcrite en texte (voir exemple 5).

Exemple 5 (Opérateur de transcription) *Pour les requêtes de la section 2, supposons que les attributs numériques sont agrégés par la moyenne Avg et les non numériques par la valeur la plus fréquente $MostFreq$ alors Q1, Q3 et Q4 s'écrivent :*

$$\mathcal{T}(\mathcal{G}_{\mathcal{I}d(Name),\mathcal{I}d(Country),\mathcal{I}d(Category),Avg(Volume),MostFreq(Methode)}(\pi_{Name,Country,Category,Volume,Methode}(Q1'))) \quad (6)$$

$$\mathcal{T}(\mathcal{G}_{MostFreq(Name),MostFreq(Country),\mathcal{I}d(Category),Avg(Volume),MostFreq(Methode)}(\pi_{Name,Country,Category,Volume,Methode}(Q3'))) \quad (7)$$

$$\mathcal{T}(\mathcal{G}_{MostFreq(Name),\mathcal{I}d(Country),MostFreq(Category),Avg(Volume),MostFreq(Methode)}(\pi_{Name,Country,Category,Volume,Methode}Q4')) \quad (8)$$

6. Implémentation et expérimentation de Stream2text

Cette section décrit notre prototype et les expérimentations réalisées pour valider l'approche. Nous abordons particulièrement la partie transcription car l'évaluation des requêtes repose sur des logiciels existants (PostgreSQL et Asteroïde (Petit, 2012)).

6.1. *Transcripteur données - texte*

Le cœur du prototype est le transcripteur développé en Java. Il prend en charge la production du résumé textuel à partir des données traitées par l'évaluateur de requêtes. Le transcripteur trie les données par entités du dictionnaire de concepts de manière à planifier la structure du document. Il se charge de récupérer les groupes de mots selon la structure grammaticale des phrases et les transfère au réalisateur de surface SimpleNLG (Gatt et Reiter, 2009). Ce dernier a la responsabilité de faire le traitement de surface des phrases et retourne les phrases correctement construites au transcripteur. Le transcripteur assemble les phrases en paragraphes pour produire un texte plus complet. La structure des comptes rendus générés actuellement comporte :
– Une introduction qui résume les informations sur lesquelles le résumé est basé. Il s'agit d'informations sur le contenu de la fenêtre : nombre de données et taille.
– Plusieurs paragraphes, un pour chaque entité du domaine d'applications (schéma des données). Chaque paragraphe est composé de plusieurs phrases dont la structure provient du dictionnaire de fonctions d'agrégats.

Le **dictionnaire de fonctions d'agrégats** utilisé pour résumer les données a été initialisé avec les fonctions suivantes :
– $MostFreq$ qui calcule la valeur la plus fréquemment rencontrée dans l'échantillon de données. Cette fonction peut être utilisée pour résumer les attributs à valeur non numérique du schéma tels que $StOpName$, $Category$, $Country$ ou $Method$.
– $\mathcal{I}d(Key\_Attribute)$ qui correspond au cas particulier évoqué dans la remarque 5.3. Par exemple, si une fonction d'agrégat $identité$ est utilisée pour résumer l'attribut $StOpName$ qui est une clef de la relation $StockOption$. Le dictionnaire contient une structure de phrases pour chaque entité du schéma le de base de données.
– Avg, Med et $Count$ (avec la sémantique habituelle), calculent respectivement la valeur moyenne, la valeur médiane pour des attributs à valeur numérique ($Volume$, $Price$, $Rate$) et le nombre total de données.
– $Part(v, A_2)$ qui calcule la part, en %, d'une valeur v dans les valeurs de A_2.

Hormis pour $\mathcal{I}d(Key\_Attribute)$, les entrées du dictionnaire contiennent une structure de phrases générique qui peut être utilisée quelque soit l'attribut sur lequel la fonction est appliquée. Ce dictionnaire est donc indépendant du schéma de la base de données et peut être partagé entre les applications et les utilisateurs. Les fonctions choisies peuvent aussi être personnalisées pour obtenir des résumés plus appropriés.

6.2. *Expérimentation avec des données de la bourse*

Un jeu de données, conforme à l'exemple, a été construit à partir de données réelles (http ://www.abcbourse.com). Les données correspondent aux cours de douze actions de dix catégories et de trois pays. Les données sont horodatées. Cette date est utilisée comme estampille pour les flux (ie. $TTime$ et $ETime$). La quantité ($Volume$), le cours ($Price$) des transactions ainsi que la volatilité ($Rate$) sont aussi estampillés. Le

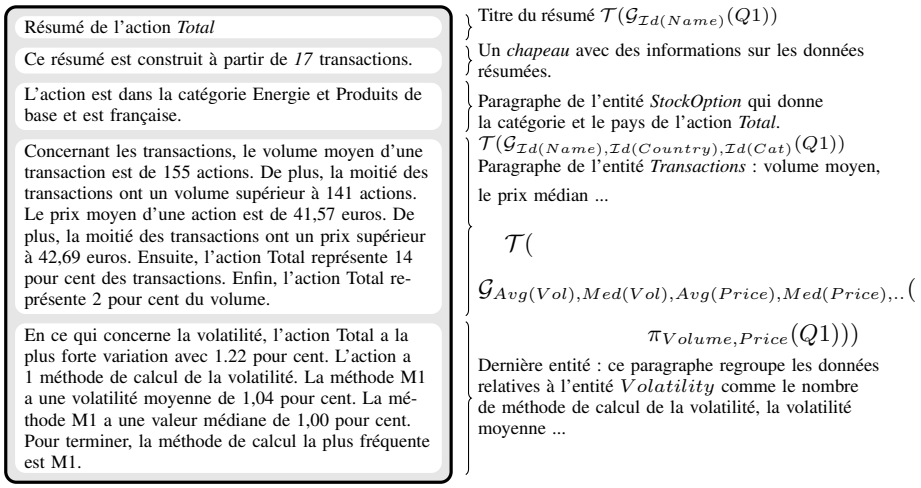


Figure 4. *Texte pour la requête Q1 selon l'exemple 6*

secteur d'activité ($Category$) et le pays ($Country$) de l'action sont disponibles. Le jeu de données ainsi constitué est composé d'environ 5000 transactions.

Le **dictionnaire de concepts** du schéma, décrit les sept concepts organisés en trois entités (cf. section 2) : l'action (attribut $StOpName$), la catégorie ($Category$), le pays ($Country$), le volume ($Volatity$), le prix ($Price$), le taux ($Rate$) et la méthode ($Method$). Pour chacun de ces concepts, une entrée comportant un groupe nominale est créée dans le dictionnaire de concepts.

Nous avons expérimenté la génération de résumés pour des requêtes analogues à celles présentés dans cet article. A titre d'illustration, nous présentons l'exemple 6.

Exemple 6 (Génération de résumé) *Pour [Q1] Luc souhaite dans son résumé la moyenne et la médiane pour le volume des transactions, pour les prix, etc. [Q1] peut s'écrire sous une forme similaire à l'équation 6. Rappelons que [Q1] analyse les données de manière continue et que le résumé sera produit successivement selon la fenêtre temporelle. La figure 4 montre les textes obtenus pour une période de 2 jours.*

Cette expérimentation nous permet de valider l'approche. A ce jour il n'y a pas eu d'expérimentations destinées à d'autres mesures de performance du système.

7. Travaux connexes

Les travaux connexes à cette proposition peuvent se grouper en trois groupes : requêtes continues sur flux, résumés numériques et "natural language generation".

Pour maîtriser les requêtes continues sur les flux de données, des travaux importants ont été réalisés tant d'un point de vue fondamental (Krishnamurthy *et al.*, 2010) que pratique (Arasu *et al.*, 2006). Dans cet article nous utilisons ASTRAL (Petit *et al.*, 2012a) qui présente l'avantage de définir de manière non-ambiguë les opérateurs sur des flux et des relations temporelles. Ceci est particulièrement important pour les jointures (Petit *et al.*, 2012b) et les fenêtres (Petit *et al.*, 2010). D'autre part, il existe de nombreux travaux sur la manière de résumer ou de synthétiser "numériquement"

des données. Dans ce contexte, on peut comprendre les modèles de préférences et les opérateurs de type $top - k$ comme un moyen de réduire la quantité de données manipulées. Notre proposition utilise des requêtes top-k mais se base aussi sur l'existence de méthodes permettant de résumer et/ou d'agréger un ensemble de valeurs en une unique valeur. Les travaux de cette nature, par exemple (Cormode *et al.*, 2012 ; Cormode et Muthukrishnan, 2005), peuvent être utilisés dans la phase d'agrégation de notre proposition. Notre proposition est suffisamment générique pour pouvoir être utilisée avec divers évaluateurs de requêtes. L'extension du langage de requêtes par un modèle de préférences (Koutrika *et al.*, 2010) n'a pas pour l'instant d'impact direct sur la transcription en langage naturel du résumé structuré. Un changement du modèle de préférences entraînerait une modification du calcul du résumé structuré mais ne modifierait pas la transcription en langage naturel. Le choix de CPrefSQL dans ASTRAL (Petit *et al.*, 2012a) est motivé par son caractère qualitatif et la possibilité de prendre en compte le "contexte" dans le calcul des n-uplets dominants. Ceci diffère des approches par fonctions de score (Borzsonyi *et al.*, 2001 ; Papadias *et al.*, 2005 ; Kontaki *et al.*, 2010).

On peut distinguer deux grandes classes d'approches pour la génération automatique de textes. L'une consiste à générer un texte à partir d'un ou plusieurs textes (*text-to-text*). Cette approche est utilisée pour résumer automatiquement des textes (Rotem, 2003) ou des opinions (Labbé et Portet, 2012). L'autre approche consiste à générer des textes qui expliquent et/ou décrivent des données (*data-to-text*). C'est dans cette dernière approche que se place notre proposition.

A notre connaissance, les travaux relevant de ce domaine, restent spécifiques à un domaine d'application. Les exemples les plus aboutis concernent la médecine (Portet *et al.*, 2009 ; Gatt *et al.*, 2009) ou la météo (Turner *et al.*, 2010). La communauté "natural language generation" travaille en particulier sur des aspects avancés du langage qui eux sont indépendants du domaine d'application ciblé : agrégation de phrases, construction de phrases énumératives, expressions référentielle,... Les phases en amont du processus d'élaboration du texte comme la détermination du contenu et la planification (Reiter et Dale, 2000) restent spécifiques au domaine d'application et nécessitent l'intervention d'experts du domaine. Cependant (Androutsopoulos *et al.*, 2013) propose une approche permettant de décrire en langage naturelle les individus ou les classes d'une ontologie OWL. Dans notre contexte, cela est assimilable à la description d'un n-uplet ou d'une relation de la base de données et non pas à la description des informations agrégées comme nous le proposons.

Dans notre approche, la détermination du contenu et la génération des phrases sont facilités puisqu'elle met à profit d'une part les connaissances conceptuelles sur la structure des données (le schéma) et d'autre part les connaissances sur les méthodes utilisées pour générer le résumé structuré des données. Les éventuelles connaissances dépendantes du domaine d'application qui sont nécessaires à la génération de texte sont capturées par le dictionnaire de schéma qui peut être élaboré lors de la description des données (par exemple lors de la spécification). Les connaissances relatives aux fonctions de résumé structuré sont elles génériques. Notre proposition met à profit la

description (la spécification) d'un ensemble de données pour en générer un résumé à l'aide d'un réalisateur de surface (Gatt et Reiter, 2009).

8. Conclusion

Notre travail s'inscrit dans un effort très actuel qui vise à maîtriser les grandes masses de données auxquelles les infrastructures informatiques doivent faire face. Nous pensons que la capacité de générer des textes courts permettant d'offrir à l'utilisateur une description synthétique de ces points d'intérêts est un atout majeur. Notre proposition prend la forme d'un système *Stream2text* qui permet de fournir un résumé en langage naturel de l'ensemble des données qui ont de l'importances pour l'utilisateur. L'approche adoptée repose sur l'utilisation des connaissances sur le schéma des données et sur la manière de les résumer. Plus particulièrement sur la manière d'exprimer en langage naturel les différentes opérations réalisées lors de la phase d'agrégation des données. Le système n'impose pas de contraintes sur les fonctions d'agrégation utilisées. L'approche proposée est indépendante de l'application visée et repose sur une intégration de bout en bout allant de requêtes utilisateur à l'expression en langage naturel des réponses. L'architecture proposée utilise des dictionnaires de concepts et de fonctions qui ouvre plus de perspectives de personnalisation pour mieux adapter le langage utilisé aux utilisateurs.

Ce travail peut être poursuivi selon de nombreux axes. Le texte généré doit pouvoir refléter les éventuelles valeurs manquantes et contenir des indications sur la fenêtre temporelle utilisée. L'aspect interface pour utilisateurs non-informaticiens serait certainement important pour faciliter l'utilisation d'un tel système. Sur les aspects données il y'a de nombreuses perspectives dont l'optimisation de l'approche, la détection d'évènements complexes et la mise à profit d'ontologies de domaines. D'autres axes de recherche sont liés à la génération automatique de texte : références à des évènements passés (ie. : *contrairement à hier*), agrégation de phrases, etc. Il est important de ne pas découpler ces différents axes de recherche car ils sont intimement liés.

9. Bibliographie

Androutsopoulos I., Lampouras G., Galanis D., « Generating Natural Language Descriptions from OWL Ontologies : the NaturalOWL System », *Journal of Artificial Intelligence Research*, vol. 48, 2013, p. 671-715.

Arasu A., Babcock B., Babu S., Cieslewicz J., Datar M., Ito K., Motwani R., Srivastava U., Widom J., « STREAM : The Stanford Data Stream Management System », Technical report, 2004, Stanford InfoLab.

Arasu A., Babu S., Widom J., « The CQL continuous query language : semantic foundations and query execution », *Proc. of 32nd int. conf. on Very Large Data bases*, vol. 15, 2006.

Borzsonyi S., Kossmann D., Stocker K., « The Skyline Operator », *Proceedings of the 17th International Conference on Data Engineering (ICDE 2001)*, 2001, p. 412–430.

Cormode G., Garofalakis M. N., Haas P. J., Jermaine C., « Synopses for Massive Data : Samples, Histograms, Wavelets, Sketches », *Foundations and Trends in Databases*, vol. 4, n° 1-3, 2012, p. 1-294.

Cormode G., Muthukrishnan S., « An improved data stream summary : the count-min sketch and its applications », *J. Algorithms*, vol. 55, n° 1, 2005, p. 58-75.

de Amo S., Pereira F., « Evaluation of Conditional Preference Queries. », *Journal of Information and Data Management (JIDM). Proceedings of the 25th Brazilian Symposium on Databases, 2010, Belo Horizonte, Brazil.*, vol. 1(3), 2010, p. 521-536.

Gatt A., Portet F., Reiter E., Hunter J., Mahamood S., Moncur W., Sripada S., « From data to text in the neonatal intensive care unit : Using NLG technology for decision support and information management », *AI Communications*, vol. 22, n° 3, 2009, p. 153-186.

Gatt A., Reiter E., « SimpleNLG : A Realisation Engine for Practical Applications », *Proceedings of the 12th European Workshop on Natural Language Generation*, ENLG '09, Stroudsburg, PA, USA, 2009, Association for Computational Linguistics, p. 90–93.

Kontaki M., Papadopoulos A., Manolopoulos Y., « Continuous Processing of Preference Queries on Data Streams », *Proc. of the 36th Int. Conf. on Current Trends in Theory and Practice of Computer Science (SOFSEM 2010)*, Springer, 2010, p. 47-60.

Koutrika G., Pitoura E., Stefanidis K., « Representation, composition and application of preferences in databases. », *Proc. of Int. Conf. on Data Engineering*, 2010, p. 1214-1215.

Krishnamurthy S., Franklin M., Davis J., Farina D., Golovko P., Li A., Thombre N., « Continuous analytics over discontinuous streams », *SIGMOD '10 : Proc. of the 2010 ACM SIGMOD int. conf. on Management of data*, ACM, 2010, p. 1081–1092.

Labbé C., Portet F., « Towards an Abstractive Opinion Summarisation of Multiple Reviews in the Tourism Domain », *The First International Workshop on Sentiment Discovery from Affective Data (SDAD 2012)*, Bristol, UK, sep 2012, p. 87–94.

Papadias D., Tao Y., Fu G., Seeger B., « Progressive Skyline Computation in Database Systems », *ACM Transactions on Database Systems*, vol. 30, 2005, p. 41–82.

Petit L., Labbé C., Roncancio C. L., « An Algebric Window Model for Data Stream Management », *Proceedings of the 9th Int. ACM Workshop on Data Engineering for Wireless and Mobile Access (MobiDE '10)* , ACM, 2010, p. 17–24.

Petit L., de Amo S., Roncancio C., Labbé C., « Top-k Context-Aware Queries on Streams », *Proc. of Int. Conf. on Database and Expert Systems Applications*, 2012, p. 397-411.

Petit L., Labbé C., Roncancio C. L., « Revisiting Formal Ordering in Data Stream Querying », *Proc. of the 2012 ACM Symp. on Applied Computing*, New York, NY, USA, 2012, ACM.

Petit L., « Gestion de flux de donnés pour l'observation de systèmes », Thèse de doctorat, Université de Grenoble, Décembre 2012.

Portet F., Reiter E., Gatt A., Hunter J., Sripada S., Freer Y., Sykes C., « Automatic Generation of Textual Summaries from Neonatal Intensive Care Data », *Artificial Intelligence*, vol. 173, n° 7–8, 2009, p. 789–816.

Reiter E., Dale R., *Building Natural Language Generation Systems*, Cambridge University Press, New York, NY, USA, 2000.

Rotem N., « Open text summarizer (OTS) », online, 2003, June, 2012, http ://libots.sourceforge.net.

Turner R., Sripada S., Reiter E., « Generating Approximate Geographic Descriptions », Krahmer E., Theune M., Eds., *Empirical Methods in Natural Language Generation*, vol. 5790 de *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, 2010, p. 121-140.

Transformer les Open Data brutes en graphes enrichis en vue d'une intégration dans les systèmes OLAP

Alain Berro[1], Imen Megdiche [2], Olivier Teste [3]

(1) *Manufacture des Tabacs, Université Toulouse I Capitole*
2 rue du Doyen Gabriel Marty, 31042 Toulouse, France

(2) *IRIT, Université Toulouse III Paul Sabatier*
118 route de Narbonne, 31062 Toulouse, France

(3) *IUT de Blagnac, Université Toulouse II Le Mirail*
1 place Georges Brassens, 31703 Blagnac, France

Alain.Berro@irit.fr, Imen.Megdiche@irit.fr, Olivier.Teste@irit.fr

RÉSUMÉ. *L'intégration des Open Data dans les systèmes OLAP est difficile en raison de l'absence de schémas sources, l'aspect brut des données et l'hétérogénéité sémantique et structurelle. La plupart des travaux existants s'intéressent aux Open Data de format RDF qui restent actuellement minoritairement disponibles. En revanche, peu de travaux s'intéressent aux Open Data de format brut, par exemple Excel qui représentent pourtant plus que 90% des données ouvertes disponibles.*
Dans cet article, nous proposons un processus automatique de transformation des Open Data brutes en graphes enrichis exploitables pour l'intégration. Ce processus est validé par l'utilisateur et s'inscrit dans notre démarche d'intégration des Open Data dans les entrepôts de données multidimensionnelles.

ABSTRACT. *The Open Data integration in the decision systems is challenged by the absence of schema, the raw data and the semantic and structural heterogeneousness. In the literature, the most of authors studies the integration of RDF'Open Data in information systems besides the little percentage of available data in this format. On the other hand, few works are interested of Excel'Open Data despite they represent more than 90% of the available data.*

In this paper, we provide an automatic process that transforms raw Open Data in exploitable rich graphs. This process is validated by the users. This is part of our generic approach for integrating the Open Data into multidimensional data warehouse.

MOTS-CLÉS : *Open Data, Entrepôts de données, Classification Conceptuelle, Graphe.*

KEYWORDS: *Open Data, Data warehouse, Conceptual Classification, Graph.*

1. Introduction

Les Open Data sont des données disponibles sous licence libre mises à disposition par des organismes publics tels que les administrations. L'ouverture de ces données a comme principal objectif la réutilisation afin de développer de nouveaux usages et de nouvelles exploitations à ces données. Notre objectif est d'exploiter ces données en les intégrant dans les systèmes décisionnels

En considérant les caractéristiques des Open Data, nous constatons qu'elles sont très riches en informations brutes. Ces informations sont précieuses pour enrichir les processus d'analyse de données (Ravat, *et al*., 2001). Toutefois ces données ont plusieurs problèmes : hétérogénéité de formats, dispersion sur une multitude de sources, métadonnées non standardisées, absence de schémas, hétérogénéité structurelle avec différents niveaux d'agrégation et imperfection des données, hétérogénéité sémantique avec des vocabulaires non communs. Nous étudions en particulier la résolution d'une partie de ces problèmes sur les Open Data de format Excel.

Les Open Data de format Excel disponibles comportent tous les problèmes cités ci-dessus. Ce type de sources correspond à la majorité des Open Data disponibles actuellement ; à titre d'exemple le site data.gouv.fr détient 300 000 fichiers dont 98,72% sont des fichiers au format Excel. Ce format intéresse depuis quelques temps la communauté scientifique (Coletta *et al.*, 2012 ; Seligman *et al.*, 2010). Plusieurs outils industriels (GoogleRefine, 2013; FusionTables, 2013) sont également proposés. Ces propositions sont intéressantes mais aucune d'elles ne traite le problème d'intégration des sources de données dans les systèmes décisionnels.

Nos travaux s'inscrivent dans le cadre des systèmes décisionnels à base d'entrepôts de données chargés de collecter et conserver les données décisionnelles. La construction d'un entrepôt de données repose sur une organisation multidimensionnelle des données entreposées (Teste, 2009). Il s'agit d'une organisation des données en sujets d'analyse appelés **faits** en fonction d'axes d'analyses appelés **dimensions**. Les dimensions sont composées de **paramètres** (ou niveau de dimension) représentant les différents niveaux de granularité possibles des axes d'analyse. Les paramètres sont organisés au sein d'une **hiérarchie**.

Dans ce contexte, notre problématique consiste à rendre les Open Data brutes exploitables dans les systèmes décisionnels par des utilisateurs. L'approche qui répond à notre problématique doit être de type Open Business Intelligence (Open BI) (Schneider *et al.*, 2011 ; Mazon *et al.*,2012) et hybride (Schneider *et al.*,2011). Une approche Open BI exige des mécanismes permettant l'extraction et l'intégration de sources hétérogènes et non structurées par des utilisateurs non-experts en BI. Une approche hybride de conception d'entrepôts est un compromis entre les données des sources et les besoins des utilisateurs pour définir le schéma multidimensionnel de l'entrepôt.

L'approche que nous proposons s'adresse à des utilisateurs chargés de concevoir l'entrepôt de données multidimensionnelles à partir d'une phase ETL semi-

automatisée pour l'intégration des Open Data sources. La définition du schéma multidimensionnel est guidée par des graphes conceptuels. L'utilisation des graphes nous permet d'une part d'assurer une évolutivité de la phase ETL (Schneider *et al.*, 2011), d'autre part une plus grande flexibilité pour l'intégration des données hétérogènes (Schneider *et al.*,2011). La phase ETL de notre approche se distingue par rapport à un outil industriel comme (Talend, 2014) par des algorithmes permettant la détection semi-automatique de données structurelles complexes et hétérogènes. Avec Talend l'utilisateur doit spécifier où se situent les données structurelles dans les fichiers.

Cet article est organisé comme suit. Nous présentons, dans la section 2, notre processus d'intégration des Open Data dans les entrepôts de données multidimensionnelles. La section 3 est dédiée à la présentation de la phase de préparation des données et la phase de définition des schémas des sources en graphes enrichis. La section 4 conclut en dressant les perspectives de ce travail.

2. Un processus d'entreposage des Open Data

Nous proposons dans la Figure 1 notre processus d'intégration des Open Data dans les entrepôts de données multidimensionnelles. Ce processus est composé de quatre phases. Il prend en entrée des sources Open Data brutes et génère en sortie un schéma multidimensionnel.

– **phase 1 : préparation des sources de données.** L'objectif de cette phase est de transformer automatiquement les données brutes en données annotées par leur type et enrichies sémantiquement. Dans cette phase nous détectons les données de type *Valeurs* contenues dans le corps des tableaux et les données de type *Structures* représentant les entêtes lignes et colonnes des tableaux comme le montre la Figure 2. Nous automatisons également la détection des données spatio-temporelles qui sont couramment utilisées dans les systèmes décisionnels. Par la suite, nous réalisons des classifications conceptuelles des données de type *Structures* afin d'enrichir les données. Enfin, l'utilisateur vérifie les détections automatiques pour garantir la validité des annotations.

– **phase 2 : définition des schémas des sources en graphes.** L'objectif de cette phase est de fusionner dans un graphe les données issues de la classification conceptuelle et les détections structurelles et spatio-temporelles. Chaque graphe enrichi obtenu représente le schéma unitaire d'un Open Data source ;

– **phase 3 : intégration des schémas en graphes des sources.** Cette phase prend en entrée un ensemble de graphes enrichis et produit un graphe intégré. Ce dernier doit vérifier un certain nombre de contraintes permettant d'assurer lors de la construction du schéma multidimensionnel, des hiérarchies strictes et couvrantes (Malinowski et Zimanyi; 2006, Hachicha, 2012) ;

– **phase 4 : définition incrémentale et semi-automatique par l'utilisateur des composants multidimensionnels.** A partir du graphe intégré des sources, l'utilisateur définit incrémentalement un schéma multidimensionnel. Ce dernier comporte d'une part les dimensions, les niveaux de dimensions et les hiérarchies,

d'autre part, l'utilisateur définit les mesures d'analyse dans le fait. Enfin, nous alimentons l'entrepôt avec les données de type *Valeurs* indexées par les instances du schéma multidimensionnel.

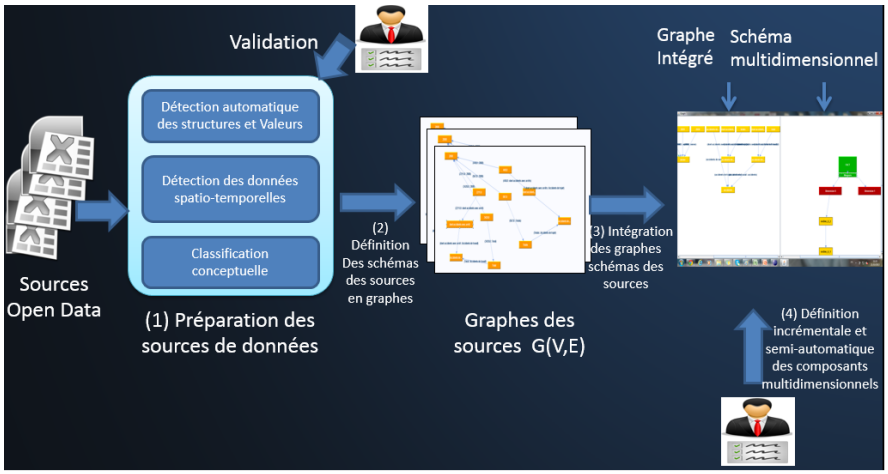


Figure 1. Processus d'entreposage des Open Data

Nous soulignons que la transformation des Open Data brutes en graphes (phase 1 et 2) peut éventuellement converger vers les solutions Linked Open Data (Böhm *et al.*, 2012) en appliquant les principes du LOD sur les graphes.

Dans ce papier, nous nous focalisons sur les phases 1 à 2 de notre processus. Nous présentons en détails dans la section 3 comment nous transformons les données Open Data brutes en graphes unitaires.

3. Transformation des Open Data brutes en graphes enrichis

Les Open Data brutes sont très riches en informations qui pourront faire l'objet d'analyses ou enrichir des analyses. Cependant elles sont difficilement exploitables dans leur format d'origine. Afin de rendre ces données exploitables nous proposons la transformation des Open Data brutes en graphe.

– un Open Data brutes contient des données de type *Valeurs* (cadre jaune) indexées par des données de type *Structures*. Ces dernières sont les concepts des entêtes lignes (cadre bleu vertical) et des entêtes colonnes (cadre bleu horizontal) (Wang, 1996) comme le montre la Figure 2 ;

– un graphe enrichi noté représente les relations entre les données de type *Structures* et les données de type *Valeurs*. Nous décrivons en détail les graphes enrichis dans la section 3.4.

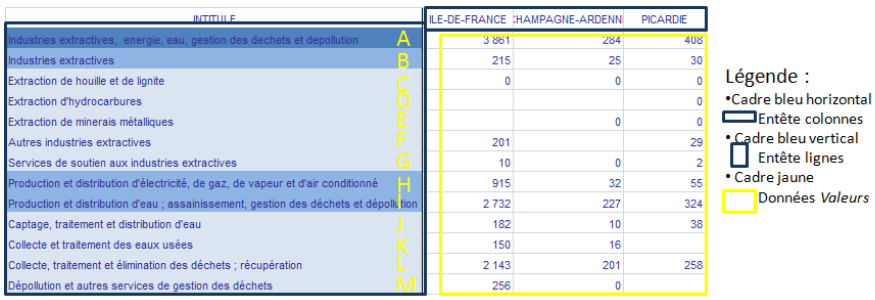


Figure 2. Un exemple d'Open Data annoté avec les types de données

3.1. Extraction des Structures et Valeurs des Open Data brutes

L'extraction des données de type *Structures* et *Valeurs* passe par cinq étapes. Dans la première étape, nous transformons les fichiers sources en matrices d'entiers en fonction du type des données dans les cellules des fichiers. Dans la deuxième étape, nous réalisons un pré-traitement sur les données brutes par exemple pour détecter les données calculées à partir d'autres données. Dans la troisième étape, nous appliquons un algorithme pour la recherche des blocs de données *Valeurs*. Dans la quatrième étape, nous faisons appel à des algorithmes de recherche des entêtes lignes et colonnes en raisonnant sur les blocs de données *Valeurs* trouvés dans l'étape 3. Enfin dans la cinquième étape, nous recherchons les blocs de données *Valeurs* similaires afin de préparer la classification conceptuelle voir section 3.3. Nous détaillons dans ce qui suit chacune des étapes.

3.1.1. Transformation des fichiers sources en matrices

Nous transformons les fichiers Open Data en matrice M d'entiers de taille $nbLigne \times nbColonne$. La matrice représente le type d'information (donnée, formule, etc.) contenu dans chaque cellule. Pour ce faire, le codage qui a été retenu est celui du Tableau 1. Par exemple une cellule qui contient une données de type *date* sera représentée dans la matrice M par l'entier 7. Les types des cellules sont détectés automatiquement. La matrice Mest définie comme suit :

$$M = (a_{i,j})_{1 \le i \le nbLigne, 1 \le j \le nbColonne} \; tel \; que \; a_{i,j} \in \{-1,0,1 \dots 9\}$$

Tableau 1. Les différents types de cellules

Type de cellules	Valeur de $a_{(i,j)}$
Vide	-1
Label	0
Numérique	1
Formule_numérique	2
Formule_label	3
Formule_erreur	4
Booléan	5
Formule_Booléan	6
Date	7
Formule_Date	8
Erreur	9

La transformation matricielle nous permet d'utiliser les types de cellules pour : découvrir automatiquement les blocs de données *Valeurs* (de type de cellules *numérique* codées par «1»), détecter les données *Structures* (de type de cellules *label* codées par «0»), détecter la présence de données temporelles, et détecter les données de type *formule*.

3.1.2. Vérification algorithmique des données de type Formule

La vérification des données de type *Formule* consiste à inspecter algorithmiquement la présence de données calculées à partir d'autres données existantes par des opérateurs tels que: avg, min, max,…. La transformation des Open Data en matrice nous permet de faire une première vérification automatique de la présence de ces dernières. En revanche, cela est insuffisant vu l'imperfection des Open Data. Ce qui nous amène à une deuxième phase de vérification qui consiste à analyser la relation entre les cellules impliquées dans le calcul d'une formule et les opérateurs qui en découlent (sum, avg, max,..).

3.1.3. Recherche des blocs de données de type Valeurs

L'objectif de cette phase est de rechercher dans la matrice *M* les blocs de données numériques, ces blocs représentent les données *Valeurs* des fichiers sources. Ces derniers sont désignés par **BlocNumUnit** (1). Ces blocs constituent des sous matrices de *M* où les indices de début et de fin de lignes et de colonnes de chaque bloc sont notés respectivement comme suit : Ligne Début (**LD**), Ligne Fin (**LF**), Colonne Début (**CD**), Colonne Fin (**CF**). L'algorithme 1 effectue la rechercher de tous les blocs de *Valeurs* contenus dans les sources de données. Le

principe de l'algorithme est de rechercher une succession de lignes qui contiennent des données numériques codées par «1» dans *M* ce qui correspond à la ligne 5 de l'algorithme. La fonction ***LignesDebEtFinBloc*** s'arrête à la première occurrence de données numériques trouvée ce qui garantit un cadrage rapide de la zone de recherche située entre LD et LF. Ainsi, dans ce cadre-là nous cherchons le(s) bloc(s) unitaire(s) numérique(s) qui se trouve(nt) dans cette zone en effectuant une vérification sur le contenu de toutes les colonnes, ce qui correspond aux lignes 8 et 12 de l'algorithme 1 en faisant appel à la fonction ***ColonnesDebEtFinBloc*.**

(1)

Algorithme 1. Recherche des Blocs de Valeurs

```
1:  Recherche Blocs Valeurs
2:  {
3:  listeBlocsNumUnit ← ∅
4:  Tant que cptLigne < nbLigne faire
5:      blocNumUnit(i) ← LignesDebEtFinBloc(cptLigne)
6:      cptColonne ← 0
7:      cptLigne ← blocNumUnit(i).LF+1
8:      Tant que cptColonne < nbColonne faire
9:          blocNumUnit(i) ← ColonnesDebEtFinBloc(cptColonne, blocNumUnit)
10:         cptColonne ← blocNumUnit(i).CF +1
11:         listeBlocsNumUnit ← listeBlocsNumUnit ∪ { blocNumUnit(i)}
12:     Fin Tant que
13:     i ← i +1
14: Fin Tant que
15: }
```

LignesDebEtFinBloc*(cptLigne) est une fonction qui renvoie la ligne de début blocNumUnit.LD et la ligne de fin blocNumUnit.LF du premier bloc de valeurs numériques détecté dans la matrice M à partir de la ligne cptLigne.*

ColonnesDebEtFinBloc*(cptColonne, blocNumUnit) est une fonction qui cherche les sous blocs contenus dans le blocNumUnit entre les lignes blocNumUnit.LD et blocNumUnit.LF et à partir de la colonne cptColonne .Cette fonction renvoie la colonne début blocNumUnit.CD et colonne fin blocTemp.CF de chaque sous bloc trouvé dans le blocNumUnit*

3.1.4. Recherche des données de type Structures

Les données de type *Structures* sont les entêtes lignes et colonnes associées à chaque bloc de *Valeurs*. Nous considérons dans nos algorithmes de recherche le fait qu'un

bloc de *Valeurs* peut être décrit soit par une entête ligne, soit par une entête colonne, soit par les deux.

Nous désignons par LEC l'indice de la ligne de l'entête des colonnes et CEL l'indice de la colonne de l'entête des lignes.

− L'entête des colonnes d'un bloc de *Valeurs k* (noté blocNumUnit(k)) est défini comme suit :

$$EnteteColonnes(k) = (a_{LEC,j})_{blocNumUnit(k).CD \leq j \leq blocNumUnit(k).CF}$$

− L'entête des lignes d'un bloc de *Valeurs k* (noté blocNumUnit(k)) est défini comme suit :

$$EnteteLignes(k) = (a_{i,CEL})_{blocNumUnit(k).LD \leq i \leq blocNumUnit(k).LF}$$

Le principe de l'algorithme que nous utilisons pour identifier le vecteur EnteteColonnes(k) (respectivement EnteteLignes(k)) de chaque bloc de *Valeurs* BlocNumUnit(k) consiste à chercher dans la matrice *M* l'occurrence de la première ligne (respectivement colonne) de 0 (correspondant à des données de type Label) se trouvant au-dessus (respectivement à gauche) du BlocNumUnit(k).

3.1.5. Recherche des blocs similaires

Les blocs similaires sont des BlocNumUnit disjoints ayant soit la même entête des lignes notés BlocSimL comme le montre la définition (2), soit la même entête des colonnes notés BlocSimC comme le montre la définition (3).

$$\begin{gathered} BlocSimC = \bigcup_{1 \leq k \leq nbBlocSim} BlocNumUnit(k) \\ tel\ que \\ CD_l = \min_k\{CD_k\} \\ CF_l = \max_k\{CF_k\} \\ EnteteColonne(k) = EnteteColonne(l) \forall k \neq l, 1 \leq k, l \leq nbBlocSim \end{gathered} \quad (2)$$

$$\begin{gathered} BlocSimL = \bigcup_{1 \leq k \leq nbBlocSim} BlocNumUnit(k) \\ tel\ que \\ LD_l = \min_k\{LD_k\} \\ LF_l = \max_k\{LF_k\} \\ EnteteLignes(k) = EnteteLignes(l) \forall k \neq l, 1 \leq k, l \leq nbBlocSim \end{gathered} \quad (3)$$

3.2. Extraction des données spatio-temporelles

Plusieurs travaux dans la littérature traitent le problème d'extraction des données spatio-temporelles (Plumejeaud, 2011; Noel et Servigne, 2006). Dans la plupart de ces travaux, l'idée est de capturer des données spatio-temporelles évolutives dans le temps. Toutefois, les données spatio-temporelles dans les Open Data ne présentent pas cet aspect. Ce qui nous amène à choisir une solution plus simple qui consiste à définir deux graphes génériques pour les entités spatio-temporelles (qui s'apparentent de manière simplifiée au mécanisme d'ontologie).

Le premier graphe générique est celui des données temporelles, il s'agit d'un graphe orienté couvrant où les nœuds sont les entités temporelles (tel que Année, Mois, Trimestre…) et l'orientation des arcs est définie du niveau le moins détaillé vers le niveau le plus détaillé ; par exemple un mois est contenu dans un trimestre, on définit alors l'orientation Mois → Trimestre. Nous utilisons le graphe temporel défini par (Mansmann et Scholl, 2007, 37). Nous proposons de compléter ce graphe en ajoutant des arcs orientés de chaque niveau inférieur vers les niveaux les plus hauts. L'objectif est de trouver toujours des chemins entre les entités temporelles que nous découvrirons dans les Open Data.

Le deuxième graphe générique est celui des entités spatiales. Nous proposons que le graphe représente le découpage géographique concerné par les Open Data que nous analysons et qu'il soit complet. Le graphe que nous utilisons pour le découpage géographique de la France comporte les entités spatiales suivantes : commune → canton → arrondissement → département → région. Ce type de graphe peut être construit de manière automatique à partir de sources de données de référence décrivant l'espace géographique souhaité ; nous ne présentons pas ce processus de construction dans cet article.

Ces graphes génériques nous permettent d'annoter les données spatio-temporelles en fonction de l'entité à laquelle elles appartiennent. Pour les entités temporelles, nous utilisons les expressions régulières pour identifier les instances de chaque entité. Par exemple l'expression suivante permet de capturer les instances des années.

Pour les entités spatiales nous utilisons des listes[1] prédéfinies d'instances ou la base GeoNames[2].

3.3. Classification conceptuelle

Les données de type *Structures* sont souvent aplaties dans les entêtes lignes et colonnes présentes dans les Open Data. A titre d'exemple dans la Figure 2, nous pouvons remarquer différents intitulés de fonction d'intérim regroupés dans la même entête ligne mais représentant différents niveaux conceptuels (ce qui correspond à différentes nuances de couleur bleu). En analysant les données indexées par ces dernières, en particulier les totaux, nous obtenons la classification conceptuelle de la Figure 3. Dans cette section, nous présentons deux stratégies de classification conceptuelle exacte et approximative soumises toutes les deux à des contraintes particulières afin de transformer les données plates de type *Structures* en hiérarchies de concepts. Nous soulignons qu'à l'issue des deux classifications, la validation des utilisateurs est souhaitable vu que l'utilisation de la sémantique et l'imperfection des données des sources peuvent fausser les résultats escomptés.

[1] http://www.insee.fr/fr/methodes/nomenclatures/cog/telechargement.asp
[2] http://www.geonames.org/

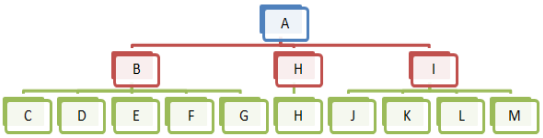


Figure 3. Classification conceptuelle des entêtes lignes

3.3.1. Les hiérarchies complexes dans les systèmes décisionnels

Un problème récurrent dans les systèmes décisionnels est la détection des hiérarchies complexes et leur impact sur les problèmes d'additivité (Mazón *et al.*, 2009). Dans la littérature, ce problème n'était pas considéré dans la phase de définition de schéma des sources (Maiz *et al.*, 2008). Toutefois, nous le trouvons bien étudié soit dans la phase d'intégration (Pedersen *et al.*, 1999) ou en temps réel dans la phase d'analyse (Hachicha, 2012). Vu les différentes difficultés que posent avec ce problème, nous avons choisi de traiter la présence de hiérarchies complexes au plus tôt dans notre démarche d'entreposage des Open Data afin d'éviter les problèmes d'additivité que nous pouvons rencontrer lors de la phase d'analyse.

Les hiérarchies sont complexes quand elles sont non-strictes, non-couvrantes ou non-strictes et non couvrantes. Nous rappelons qu'une hiérarchie est composée de paramètres.

- une hiérarchie est non-stricte (Malinowski et Zimanyi, 2006) si un paramètre fils a plus qu'un parent. Par exemple, un film A fait partie des deux catégories de films « science-fiction » et « tragédie »;
- une hiérarchie est non-couvrante (Malinowski et Zimanyi, 2006) si certains paramètres de la hiérarchie n'ont pas d'instances. Par exemple, dans la hiérarchie « magasin-ville-région-pays » un magasin peut être associé à une région sans être affecté à une ville;
- une hiérarchie non-ontologique ou non-équilibrée est un cas particulier de hiérarchie non-couvrante qui comporte des instances manquantes pour les paramètres de niveau feuille. Par exemple dans la hiérarchie « magasin-ville-région-pays », une ville peut ne pas héberger de magasin.

Dans la section suivante, nous expliquons comment nous interdisons ces types de hiérarchies sous la forme de contraintes imposées au processus de classification conceptuelle.

3.3.2. Les contraintes du processus de classification

Le processus de classification conceptuelle doit vérifier les contraintes C1, C2 et C3. Ce processus prend en entrée des données de type *Structure* (non spatio-temporelles) et produit en sortie des hiérarchies structurelles (englobant schéma et instance) sous format d'arbre. Les contraintes sont les suivantes:

- C1 : Pour chaque feuille *fi* de l'arbre(k), le chemin entre *fi* et la racine de l'arbre(k) est unique. Ce qui signifie que chaque nœud de l'arbre à l'exception

de la racine a exactement un seul parent. Cette condition permet de garantir des hiérarchies strictes à l'échelle du schéma de la source de données.

– C2 : Si un nœud *n* dans l'arbre à l'exception des racines n'a pas de parent ou un parent qui n'a pas de fils, nous dupliquons le nœud *n* dans le niveau manquant. Cette condition permet de garantir des hiérarchies couvrantes au niveau du schéma des sources.

– C3 : La hauteur de l'arbre doit être identique en partant de n'importe quelle feuille vers la racine de l'arbre. Cette condition permet de garantir des hiérarchies ontologiques au niveau du schéma de la source de données.

3.3.3. Classification conceptuelle exacte

Dans plusieurs fichiers Open Data, l'organisation des données pourrait indiquer la classification conceptuelle des données *Structures*. Par exemple, la disposition des blocs de données *Valeurs*, les données de type *Formule*, les cellules fusionnées sont des indicateurs de classification conceptuelle. Nous proposons dans cette section une première stratégie de classification conceptuelle exacte. Pour cette stratégie, nous avons en entrée les données de type *Structures* qui sont les concepts à classifier et en sortie nous générons des arbres (ou hiérarchies) de concepts vérifiant les contraintes C1, C2 et C3. Plusieurs algorithmes sont proposés :

(1) **ClassifConceptEntêtesLignes** est un algorithme qui raisonne sur la disposition des blocs de données *Valeurs* pour classifier les concepts de l'entête des lignes. La Figure 4 décrit à droite l'arbre résultant de l'application de cet algorithme. Ce dernier permet d'affecter les concepts entre les BlocNumUnit(k) au niveau 1. Par exemple pour le premier bloc de *Valeurs* (en jaune) à gauche dans la Figure 4 les deux concepts « Ecole maternelle » et « Ecole élémentaire » sont de niveau 1 ce qui correspond à la partie entre les lignes 6 à 11 de l'algorithme 2. Le concept « Enseignement public » est de niveau 2 et « Premier degré » est de niveau 3 ce qui correspond à la partie entre les lignes 12 à 18 dans l'algorithme 2.

(2) Un deuxième algorithme associé à la section 3.1.2 permet classifier les entêtes lignes ou colonnes qui indexent des lignes ou colonnes de données de type *Formules.* La relation entre les formules se traduit par un arbre entre les concepts de l'entête lignes ou colonnes. La Figure 3 illustre un exemple de cet algorithme, le concept H est dupliqué pour vérifier les contraintes C1, C2 et C3.

(3) Un algorithme pour la recherche des cellules fusionnées au-dessus des entêtes colonnes (respectivement à gauche des entêtes lignes) qui permet de construire un arbre tel que chaque cellule fusionnée est le parent des concepts fils au-dessous (respectivement à droite) de cette dernière.

Algorithme 2. Classification conceptuelle des concepts de l'entête des lignes

---

1: **ClassifConceptEntêtesLignes(BlocSimL)**
2: {
3: $k \leftarrow 1$

```
4:      sousBlocCrt ← BlocNumUnit(k)
5:      Tant que (i < BlocSimL.LF ET k < nbrBloc ) faire
6:       Si  (i =sousBlocCrt.LD ) alors
7:         Affecter tous les concepts entre  sousBlocCrt.LD et sousBlocCRt.LF à Niv1
8:         k ← k+1
9:         sousBlocCrt ← BlocNumUnit(k)
10:        i ← sousBlocCrt.LF + 1
11:      Fin Si
12:      Si ( i < sousBlocCrt.LD) alors
13:        Compter le nbreConcept  de type labels entre la ligne i et sousBlocCrt.LD
14:        Pour j de i à sousBlocCrt.LD faire
15:          Affecter à chaque concept label le niveau nbreConcept+1
16:          Décrémenter nbreConcept
17:        Fin Pour
18:       Fin Si
19:   Fin Tant que
20:}
```

	2005-2006	2006-2007	2007-2008	2008-2009	2009-2010
Premier degré					
Enseignement public					
Écoles maternelles	26,1	25,9	25,8	25,8	25,5
Écoles élémentaires	22,5	22,6	22,7	22,6	22,6
Enseignement privé					
Écoles maternelles	27,6	27,3	26,9	26,7	27,1
Écoles élémentaires	22,8	22,9	23,1	23,0	22,9
Second degré					
Enseignement public (1)					
1er cycle (2)	24,0	24,0	23,9	24,1	24,1
2e cycle professionnel	20,1	19,7	19,4	19,0	19,0
2e cycle général et technologique	26,8	26,5	26,3	26,4	26,4
Enseignement privé					
1er cycle (2)	24,7	24,8	25,0	25,0	25,2
2e cycle professionnel	19,2	19,1	18,8	18,3	18,6
2e cycle général et technologique	25,0	24,9	24,9	24,8	25,1

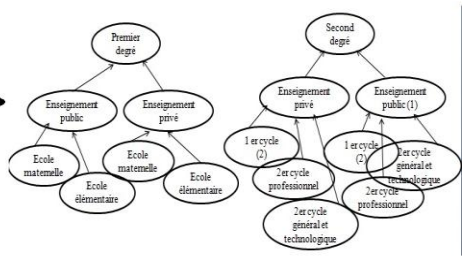


Figure 4. Exemple de classification exacte

3.3.4. Classification conceptuelle approximative

Dans la section *3.3.2*, nous avons présenté une approche de classification conceptuelle exacte qui exige la présence d'indicateurs d'organisation. Toutefois, vu l'imperfection des données présentes dans les Open Data, la première approche n'est pas toujours applicable ce qui nous amène à proposer une deuxième approche, plus souple, de classification conceptuelle approximative. Dans l'approche approximative, nous croisons les résultats de classification de la technique des treillis de gallois (Birkho, 1967) avec les résultats de l'approche RELEVANT (Bergamaschi *et al*., 2007) sous les contraintes C1, C2 et C3 pour pouvoir transformer un ensemble de données de type *Structures* qui sont les concepts à classifier en arbre à deux niveaux.

La classification conceptuelle avec les treillis de galois produit des contextes formels à plusieurs attributs. Cette technique ne prend pas en compte les aspects

sémantiques des concepts et les hiérarchies produites ne sont pas strictes. Pour les aspects sémantiques, nous avons fait appel à l'approche RELEVANT qui considère la similarité sémantique pour regrouper un ensemble de concepts en clusters représentés avec les attributs les plus pertinents selon la technique de clustering choisie. L'approche RELEVANT propose deux techniques de clustering : la première technique est hiérarchique, elle produit des clusters disjoints dont les noms sont composés de plusieurs attributs, la deuxième technique basée sur les clusters non-disjoints produits des clusters de nom mono-attribut. Nous avons choisi d'appliquer la technique de clustering hiérarchique afin d'obtenir des clusters disjoints plus proches de la contrainte C1. Nous proposons dans ce qui suit d'illustrer notre approche de classification sur les concepts de l'entête des lignes de la Figure 2 :

1. Préparation des données : Pour l'ensemble des *Concepts = {A ;B; C ; D; E; F; G; H; I; J; K; L; M}*, nous faisons une extraction des racines des mots distincts , nous obtenons l'ensemble *Attributs = {(A1) industr; (A2) extract; (A3) énerg; (A4) gestion; (A5) dépollu; (A6) houill; (A7) lignit; (A8) hydorcarbur; (A9) métalliques; (A10) autr; (T11) servic; (A12) product; (A13) distribu; (A14) électr; (A15) gaz; (A16) condition; (A17) déchet; (A18) captag; (A19) trait; (A20) collect; (A21) usé; (A21) élimin} ;*

2. Sur l'ensemble des *Concepts*, nous appliquons l'algorithme de classification des treillis de galois et nous faisons l'extraction des contextes formels mono-attribut (Berro *et al*., 2013), voir Figure 5 ;

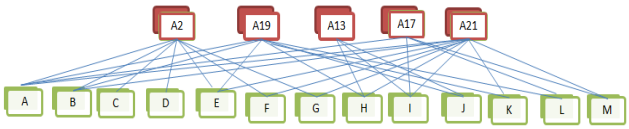


Figure 5. Contextes mono-attribut du treillis

3. Sur l'ensemble des *Concepts*, nous appliquons l'approche RELEVANT avec le clustering hiérarchique. Nous obtenons des clusters disjoints où chaque cluster est représenté par un ensemble d'attributs, voir Figure 6 ;

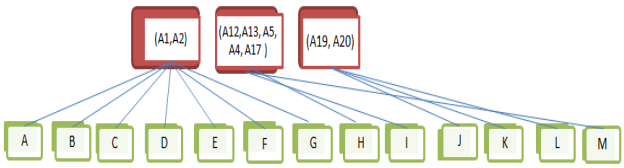


Figure 6. Clusters obtenus par RELEVANT

4. Nous sélectionnons les mono-attributs du treillis qui ont été sélectionnés comme pertinents par RELEVANT, dans notre exemple *AttriPerti = {A2, A13, A17, A19}*. Ensuite, chaque attribut sera lié à l'intersection des concepts entre les deux approches par exemple *A13* sera lié à $\{H, I\} = \{H, I, M\} \cap \{H, I, J\}$ et *A17* sera lié à $\{I, M\} = \{H, I, M\} \cap \{A, I, L, M\}$. Ceci produit des hiérarchies non-strictes, nous résolvons ce problème en

attachant le concept I à l'attribut le plus proche sémantiquement : I est plus proche sémantiquement de A13 que de A17 avec la mesure de Jaccard, voir Figure 7 pour le résultat de classification conceptuelle.

Figure 7. Exemple de classification approximative

En comparant les résultats de classification conceptuelle obtenus par l'approche exacte et l'approche approximative, nous remarquons que nous avons réussi à obtenir un sous ensemble des concepts correctement regroupés tel que les ensembles {J, K, L} ou {C, D, E, F, G}.

3.4. Règles de transformation des Open Data brutes en graphes enrichis

Un graphe enrichi, noté $G = (V, E)$, décrit les différents types de relations entre les données de type *Structures*, les concepts d'enrichissement et les données de type *Valeurs*.

L'ensemble V des sommets est partitionné en quatre sous-ensembles :

- $V_{EnteteLigne,BlocNumUnit(k)}$ l'ensemble des concepts de type *Structures* de l'entête des lignes associé au bloc de *Valeurs* (blocNumUnit (k)) ;
- $V_{EnteteColonne,BlocNumUnit(k)}$ l'ensemble des concepts de type *Structures* de l'entête des colonnes associés à des blocs de *Valeurs* (blocNumUnit(k)) ;
- $V_{StructEnrichi}$ l'ensemble des concepts associés au blocSim, issus de la classification conceptuelle externe ou des cellules fusionnées ;
- $V_{Val,BlocNumUnit(k)}$ l'ensemble des données de type *Valeurs* associées au BlocNumUnit(k).

L'ensemble E des arcs est partitionné en cinq sous-ensembles:

- $E_{StructEnrichi,StructEnrichi}$ l'ensemble des arcs représentant les liens entre les concepts d'enrichissement où $u, v \in V_{StructEnrichi}$;
- $E_{structEnrichi,EnteteLignes}$ l'ensemble des arcs représentant les liens entre les concepts d'enrichissement et les concepts de l'entêtes des lignes où ;
- $E_{StructEnrichi,EnteteColonnes}$ l'ensemble des arcs représentant les liens entre les concepts d'enrichissement et les concepts de l'entêtes de colonnes où ;
- $E_{EnteteLignes,Val,BlocNumUnit(k)}$l'ensemble des arcs représentant les liens entre les concepts de l'entête des lignes et les données de type *Valeurs* où ;

- $E_{EnteteColonnes,Val,BlocNumUnit(k)}$l'ensemble des arcs (u, v) représentant les liens entre les concepts de l'entête des colonnes et les données de type *Valeurs* où $u \in V_{EnteteColonnes,BlocNumUnit(k)}$ et $v \in V_{Val,BlocNumUnit(k)}$.

Nous présentons dans ce qui suit les règles de transformations des données extraites des Open Data brutes dans les graphes enrichis.

Règle 1 : Les données de type *Valeurs* extraites dans la section 3.1.3 sont transformées en sommets de type $V_{Val,BlocNumUnit(k)}$.
Règle 2 : Les données de type *Structures* extraites dans la section 3.1.4 sont transformées en sommets de type $V_{EnteteColonne,BlocNumUnit(k)}$ pour les concepts de l'entête des colonnes du BlocNumUnit(k) et des sommets de type $V_{EnteteLigne,BlocNumUnit(k)}$ pour les concepts de l'entête des lignes du BlocNumUnit(k).
Règle 3 : A partir des données spatio-temporelles, obtenues à la section 3.2, nous transformons les entités spatio-temporelles en sommets de type $V_{StructEnrichi}$ et les données spatio-temporelles extraites en $V_{EnteteLigne,BlocNumUnit(k)}$ ou $V_{EnteteColonne,BlocNumUnit(k)}$. Nous relions ces sommets avec des arcs de type $E_{structEnrichi,EnteteLignes}$ ou $E_{StructEnrichi,EnteteColonnes}$.
Règle 4 : A partir des arbres, obtenus à la section 3.3.3 et à la section 3.3.4, les feuilles sont les sommets obtenus à partir de la règle 2. Les autres nœuds des arbres sont transformés en sommets de type $V_{StructEnrichi}$. Nous relions ces concepts en respectant le sens des arcs dans les arbres d'origine avec des arcs de type $E_{structEnrichi,EnteteLignes}$, $E_{StructEnrichi,EnteteColonnes}$ et $E_{StructEnrichi,StructEnrichi}$

4. Conclusion

Les Open Data brutes sont actuellement difficilement exploitables dans les systèmes OLAP vu leur hétérogénéité structurelle et sémantique, leur aspect bruts (par exemple manque de hiérarchies), et les imperfections dans ces données. Nous avons proposé dans cet article un processus permettant de transformer les Open Data brutes en graphes. Les algorithmes de détection sont génériques (s'appliquent sur les fichiers contenant des données numériques et restent valables sur les autres types de fichiers). Toutefois les techniques de classification et les contraintes sous-jacentes ne peuvent pas être complétement automatisées vu l'imperfection des données traitées. Dans nos futurs travaux, nous présentons les résultats d'expérimentations de notre processus ETL sur une grande masse de fichiers Open Data brutes. Nous détaillons également la phase d'intégration des graphes enrichis dans les systèmes décisionnels à base d'entrepôts de données multidimensionnelles.

Bibliographie

Berro A., Megdiche I., Teste O. (2013). Vers l'intégration multidimensionnelle d'Open Data dans les entrepôts de données. *EDA'13*.

Birkho, G. (1967). *Lattice Theory*.

Böhm C, Freitag M, Heise A, et al. (2012). GovWILD: integrating open government data for transparency. *WWW (Companion Volume)*.

Coletta R., Castanier E., Valduriez P., Frisch C., Ngo DH., Bellahsene Z. (2012). Public Data Integration with WebSmatch, *CoRR'12*.

FusionTables. (2013). http://www.google.com/drive/apps.html#fusiontables

GoogleRefine. (2013). http://code.google.com/p/google-refine

Hachicha M. (2012). *Modélisation de hiérarchies complexes dans les entrepôts de données XML et traitement des problèmes d'additivité dans l'analyse en ligne XOLAP*. Thèse en Informatique, Université Lunière Lyon 2.

Maiz N., Boussaid O., Fadila Bentayeb F(2008). Fusion automatique des ontologies par classification hiérarchique pour la conception d'un entrepôt de données. *EGC'08*.

Malinowski E., Zimanyi E. (2006) Hierarchies in a multidimensional model: From conceptual modeling to logical representation. *Data & Knowledge Engineering*, vol 59, n°2, p 348-377.

Malú C., Florian D., Garrigós I., Mazón J-N. (2013). Business Intelligence and the Web Guest editors' introduction. *Information Systems Frontiers*, vol.15, n° 3, p. 307-309.

Mansmann S., Scholl M.H. (2007). Empowering the OLAP Technology to Support Complex Dimension Hierarchies. *IJDWM*, vol. 3, n° 4. p 31-50.

Bergamaschi S., Sartori C., Guerra F. Mirko Orsini M. (2007). Extracting Relevant Attribute Values for Improved Search. *Internet Computing, IEEE*, vol. 11, n °5, p 26-35.

Mazón, J.-N., Lechtenbörger, J. and Trujillo, J. (2009). A survey on summarizability issues in multidimensional modeling. *Data & Knowledge Engineering*, vol 68, n° 12 , p 1452-1469

Mazon J.N., Zubcoff J.J., Garrigos I., Espinosa R., Rodriguez R. (2012). Open business intelligence: on the importance of data quality awareness in user-friendly data mining. *2nd International workshop on linked web data management, LWDM'12*.

Noel G., Servigne S. (2006). Structuration de données spatio-temporelles temps-réelles : vers la gestion de la saturation de base de données. INFORSID'06.

Pedersen T.B., Jensen C.S., Dyreson C.E (1999). Extending Practical Pre-Aggregation in On Line Analytical Processing. *In 25th International Conference on Very Large Data Bases, VLDB'99*.

Plumejeaud C. (2011) *Modèles et méthodes pour l'information spatio-temporelle évolutive*. Thèse de doctorat. Université de Grenoble.

Ravat F., Teste O., Zurfluh G. (2001). Modélisation multidimensionnelle des systèmes décisionnels. *Extraction et Gestion des Connaissances (EGC'01), Revue des Sciences et Technologies de l'Information, RIA-ECA, Hermès, Vol.1, N°1-2, p.201-212,*

Seligman L., Mork P., Halevy A., Smith K., Carey M.J., Chen K., Wolf C., Madhavan J.,Kannan A. (2010). OpenII: an open source information integration toolkit. *In Int, SIGMOd Conference.*

Schneider M., Vossen G., Zimanyi E. (2011). *Data Warehousing: from occasional OLAP to real-time business intelligence*. Dagstuhl Reports, 1(9), p.1-25.

Talend. (2014). http://fr.talend.com/products/data-integration.

Teste O. (2009). *Modélisation et manipulation des systèmes OLAP : de l'intégration des documents à l'usager*. Habilitation à Diriger des recherches de l'Université Paul Sabatier Toulouse 3.

Wang X. (1996). Tabular abstraction, Editing and formatting. Phd Thesis, University of Waretloo, Waterloo, Ontaria, Canada.

Session 2b

Ingénierie des exigences : modélisation, vérification et traçabilité

Des buts à la modélisation système : une approche de modélisation des exigences centrée utilisateur

Fernando Wanderley* — **Nicolas Belloir**** — **Jean-Michel Bruel***** — **Nabil Hameurlain**** — **JoãoAraújo***

* *CITI, FCT, Universidade Nova de Lisboa, Caparica, Portugal*
(f.wanderley, p191)@fct.unl.pt
** *LIUPPA, Université de Pau et des Pays de l'Adour*
BP 1155, 64013 Pau Cedex, France
(nicolas.belloir,nabil.hameurlain)@univ-pau.fr
*** *CNRS, IRIT, Université de Toulouse, Cedex, France*
bruel@irit.fr

RÉSUMÉ. *Un des problèmes en ingénierie des exigences consiste à capter les besoins des utilisateurs le mieux possible. Or force est de constater que les supports d'ingénierie tels que les modèles orientés but ou les diagrammes d'exigences orientés système tels que ceux de* SysML *sont parfois trop complexes pour les utilisateurs finaux. Dans cet article, nous proposons un processus systématique permettant aux utilisateurs d'exprimer les exigences à l'aide de modèles cognitifs plus simples tels que les Mind Maps. Ces derniers sont alors transformés en modèles* Kaos *puis en modèles* SysML *en appliquant des techniques de transformations de modèles. Nous avons appliqué cette approche à un cas d'application industriel.*

ABSTRACT. *One of the existing problems in requirement engineering consists in arriving to capture final user requirements as well as possible. But it is clear that engineering methods such as goal-oriented models or system-oriented requirement diagrams such as* SysML *are sometimes too hard for end-users. In this paper we define a systematic process allowing ennd-users to specify requirements using cognitive models such as mind maps. Then, those models are transformed into* Kaos *models, then in* SysML *models, using model transformation rules. This approach is applied to an industrial case application.*

MOTS-CLÉS : *Modèles de recueil des exigences graphiques, ingénierie des exigences orientée utilisateurs, ingénierie des modèles, Mind Maps,* Kaos, SysML

KEYWORDS: *Modèles graphiques des exigences, Recueil des besoins centré-utilisateur, Ingénierie des modèles, Mind Maps,* Kaos, SysML

1. Introduction

Selon une étude récente menée par le Standish Group[1], environ 66 % des logiciels développés ne répondent pas aux attentes des utilisateurs que ce soit au niveau de leurs fonctionnalités (niveau système) ou au niveau de leurs comportements (niveau buts). Ce rapport indique que la plus grande part des échecs relevés était liée à des malentendus et un manque de communication entre les ingénieurs des exigences et les utilisateurs. Ces données confirment qu'aucune autre partie du travail de création d'un logiciel n'est aussi difficile que d'établir une communication fiable entre les financiers, les experts domaines, les ingénieurs des exigences et les ingénieurs logiciels (Pressman, 2006 ; Sommerville, 2010).

D'autre part, les modèles graphiques des exigences tels que les modèles orientés but (comme KAOS (Dardenne *et al.*, 1993)) ou les modèles SYSML[2] (Object Management Group, 2012) pour l'ingénierie système sont une des manières les plus efficaces pour identifier les exigences d'un système et les besoins utilisateurs.

Dans ce contexte, des études telles que (Sommerville, 2010) ou (Wanderley *et al.*, 2012) ou (Wanderley et Araujo, 2013) ont montré que des modèles cognitifs tels que les *Mind Maps*[3] pouvaient être utilisés en ingénierie des exigences afin de faciliter la communication entre les différentes parties prenantes. Ces travaux ont également montré que les techniques d'ingénierie des modèles pouvaient aider à la génération de modèles d'exigences traditionnels (tels que des modèles conceptuels exprimés en UML ou encore en KAOS). Cette approche permet de réunir les propriétés bien connues des *Mind Maps* et les techniques de transformations de modèles afin de transformer automatiquement les exigences exprimées sous la forme de *Mind Maps* en modèles logiciels tels que diagrammes de classes UML, modèles fonctionnels et modèles KAOS. Enfin, dans une étude récente (Wanderley et Araujo, 2013), nous avons montré comment l'utilisation de *Mind Maps* et des techniques agiles de recueil des exigences, encourageait la construction de modèles de buts KAOS plus efficacement et plus simplement, en impliquant des utilisateurs non formés à KAOS.

Dans ce papier, nous étendons ces techniques afin de définir des transformations permettant de passer des modèles KAOS générés aux modèles SYSML suivants : diagrammes des exigences et diagrammes de définition de blocs. Une étude de cas industrielle complète (expression des exigences en *Mind Maps*, transformation de ces modèles en KAOS, puis transformation en modèles SYSML) a été réalisée.

La suite de ce papier est organisée comme suit. La section 2 présente les concepts nécessaires à notre approche, c'est-à-dire les *Mind Maps*, les modèles orientés but (et plus particulièrement KAOS), SYSML et les techniques d'ingénierie des modèles. La section 2.5 décrit les métamodèles de chacun, et les transformations entre eux.

1. Standish Group -Accessed May 2012 - `http://blog.standishgroup.com/`
2. SYSML - System Modeling Language
3. Nous utilisons volontairement le terme *Mind Maps* au lieu de traductions françaises telles que carte heuristique ou carte cognitive ou carte mentale ou carte des idées afin de rester sur un terme générique reconnu et supporté par des logiciels tels que *xMaps* ou autres.

La section 3 décrit l'application de notre approche dans un contexte industriel et plus particulièrement les transformations entre les modèles *Mind Maps* et KAOS, puis entre KAOS et SYSML. La section 4 présente quelques approches similaires tandis que la section 5 apporte une conclusion et présente des directions pour de futurs travaux.

2. Fondements

2.1. *Les modèles Mind Maps en tant que modèles centrés utilisateur*

Une *Mind Map* selon (Buzan et Buzan, 2003) est un diagramme utilisé pour voir, classifier et organiser des concepts, ainsi que pour générer de nouvelles idées. Elle est utilisée pour connecter des mots, des idées et des concepts à une idée ou un concept central. Elle est similaire à un réseau sémantique ou à une carte cognitive, mais sans les restrictions sur les types de connections utilisées. Une *Mind Map* est un diagramme radial qui, à l'aide d'un vocabulaire (c'est à dire d'un ensemble de mots-clés), peut représenter et modéliser de manière cognitive un concept ou un domaine spécifique. Les principaux bénéfices de l'utilisation de ce type de représentation sont : organisation des idées et des concepts, mise en accent de mots significatifs, association entre éléments d'une branche, regroupement d'idées, support à la mémoire visuelle et à la créativité, déclenchement d'innovations, le tout de manière simple. Certaines études, résumées dans (Wanderley et Araujo, 2013), soulignent que ce type de représentation rend plus facile pour l'esprit humain le traitement de l'information, tout en réduisant la charge cognitive nécessaire pour absorber les concepts du domaine et leurs objectifs.

Lors de travaux précédent, nous avons mis en évidence que plusieurs techniques et outils basés sur les *Mind Maps* sont considérés, tant par le monde académique que par le monde industriel, comme de puissants outils pour l'élicitation des exigences, notamment en développement agile (Wanderley et Araujo, 2013) ; nous renvoyons à la lecture de cet article pour les détails de cette étude bibliographique.

L'ingénierie des exigences est reconnue comme un processus social qui se caractérise par des prises de décisions permanentes entre de nombreux participants (gestionnaires, utilisateurs finaux, et analystes du système). Lors de la phase de recueil des exigences, (Moody *et al.*, 2003) affirme que le développement logiciel est plus un travail artisanal qu'une réelle discipline d'ingénierie. À cet égard, la cognitive humaine joue un rôle essentiel pour la compréhension des problèmes humains et organisationnels dans l'ingénierie des exigences, et sert à identifier les moyens d'amélioration de la qualité de la perception visuelle du modèle de spécification produit. Dans cet article, nous adoptons une représentation visuelle centrée sur l'utilisateur basée sur les modèles mentaux (Davidson *et al.*, 1999).

Dans ce contexte, ce papier propose l'adoption des *Mind Maps* afin de faciliter la communication (entre les utilisateurs finaux et les ingénieurs des exigences) et offrir ainsi un support simplifié pour la modélisation des buts.

2.2. *Ingénierie des exigences basée sur les buts*

L'ingénierie des exigences basée sur les buts (GORE[4]) considère l'organisation et les objectifs des acteurs comme la source des exigences (fonctionnelles et non-fonctionnelles). Avant de travailler à l'élicitaton des exigences, il convient de se concentrer sur l'élicitation des buts (van Lamsweerde, 2001). Les GOREs modélisent donc les buts. Ces derniers peuvent être vus comme les propriétés désirées du système qui ont été exprimées par les utilisateurs. En outre, ils peuvent être spécifiés à différents niveaux d'abstraction, couvrant les préoccupations stratégiques à haut-niveau et à un niveau inférieur les problèmes techniques. Selon la classification de (van Lamsweerde, 2001), plusieurs méthodes peuvent être considérées comme appartenant aux GOREs : le framework i* (Yu, 1995), GBRAM (Antón *et al.*, 1995), NFR (Lawrence Chung, 2000), KAOS (van Lamsweerde, 2001) et (Dardenne *et al.*, 1993), TROPOS (Castro *et al.*, 2002), la carte des buts/stratégies(Rolland et Salinesi, 2005), ou encore GRL (University of Toronto,). Parmi ces méthodes, KAOS est l'une des plus citées. C'est pour cette raison que nous avons focalisé notre travail dessus.

Un *but* en KAOS peut être défini comme une déclaration d'intention sur un système dont la satisfaction, en général, exige la coopération de certains *agents* qui configurent le système. Les buts sont satisfaits par les *exigences* qui sont prises en compte dans les spécifications des opérations du logiciel ou par des hypothèses qui expriment un comportement effectué par des *agents extérieurs*. Les *agents logiciels* sont des composants actifs qui réalisent des opérations satisfaisant les exigences pour lesquelles elles ont été définies. KAOS propose quatre visions d'un problème à travers les modèles suivants : le *modèle des buts*, le *modèle objet*, le *modèle de responsabilité* et le *modèle opérationnel*. Ces modèles sont basés sur les *buts (Goal)*, les *exigences (Requirement)*, les *entités (Entity)*, les *évènements (Event)*, et les *relations (Relationship)* entre ces concepts. Des *objets* peuvent être spécifiés pour décrire le modèle structurel du projet. Ils peuvent être passifs (*entity*, *event* ou *relationship*) ou actifs (*agents*). Des obstacles (*contraintes (constraint)*) et des relations entre buts (*conflits entre buts (conflict goals)*) sont utilisés pour analyser les scenarii dans lesquels des buts pourraient être non satisfaits, et contribuent ainsi à l'identification des vulnérabilités du système (Lamsweerde et Letier, 2003).

2.3. *Le langage* SYSML

SYSML est un langage de modélisation graphique pour l'ingénierie système, et plus particulièrement pour les systèmes complexes. Il est utilisé pour spécifier, analyser, concevoir et vérifier les systèmes complexes. Ce langage fournit des représentations graphiques basées sur une sémantique semi-formelle pour modéliser les exigences, la structure, le comportement et certains aspects mathématiques d'un système. Il peut être intégré avec d'autres méthodes et modèles d'ingénierie.

4. GORE - Goal-Oriented Requirements Engineering

Ce langage a été défini comme une extension de UML 2[5]. Il est construit à partir d'un sous-ensemble de ce dernier auquel on a ajouté un certain nombre de concepts manquant afin de satisfaire l'ingénierie système. Son utilisation par le monde industriel connaît une croissance significative ; il a été utilisé dans la réalisation de cas industriels en production. Par exemple, Airbus l'a utilisé pour modéliser une partie des spécifications de l'A350 (Riviére et Benac, 2010). Dans cette étude, nous nous limitons à deux de ses neuf diagrammes.

Le *diagramme des exigences* (*Requirements Diagram*) permet de représenter les exigences et les relations entre-elles de deux manières. Soit à l'aide d'un tableau, soit à l'aide d'une représentation graphique basée sur le diagramme de classes d'UML. Dans les deux cas, les exigences sont représentées selon un point de vue décompositionnel, permettant d'exprimer la traçabilité, la vérification ou la satisfaction par ou vers des éléments autres du système (tels que des blocs structurels (*blocks*) ou des activités (*activity*) par exemple).

Le *diagramme de définition de blocs* (*Block Definition Diagram*) étend également le diagramme de classe d'UML. Il représente la vue structurelle du système d'un point de vue externe. Le système est donc vu comme un ensemble de systèmes et de sous-systèmes. Il aide à capturer les composants constituant le système et les relations entre eux. La vue interne structurelle du système est donnée par un autre diagramme, le diagramme de blocs internes (*Internal Block Diagram*), représentant la vue interne d'un bloc (spécifié dans un diagramme de définition de blocs) à l'aide de parties et de connexions entre ces parties.

D'autres diagrammes sont définis par SysML, notamment des diagrammes comportementaux, mais nous ne les considérons pas dans cette étude. La traçabilité entre éléments du système est une des avancées importantes de SysML. Elle permet notamment de relier les exigences aux éléments de modèles les satisfaisant. Cela permet d'assurer une meilleure couverture des exigences et de leur prise en compte.

2.4. *Ingénierie basée sur les modèles*

L'ingénierie basée sur les modèles (MDE[6]) se concentre sur les modèles comme principal intérêt (Schmidt, 2006). L'utilisation des modèles pour la construction de systèmes complexes est un des moyens permettant de résoudre les capacités cognitives limitées des humains à comprendre un système complexe dans sa globalité (Kleppe *et al.*, 2003). La MDE propose de réduire les besoins qu'a l'ingénieur logiciel d'interagir manuellement avec le code source, lui permettant de se concentrer sur des modèles de plus haut-niveau, dans lesquels il peut notamment s'abstraire des problèmes de développement des différentes plateformes supports. Cette stratégie permet d'améliorer certains points clés de la création d'un système tels que la productivité, l'interopérabilité et la communication (Mernik *et al.*, 2005).

5. UML - Unified Modeling Language
6. MDE : Model Based Engineering

Le développement par les modèles requiert une définition rigoureuse de ces derniers. Cela est réalisé par la définition de *métamodèles* et par la spécification de *transformations* entre ces métamodèles. Ces dernières permettent de transformer un modèle exprimé grâce à un métamodèle donné en un autre modèle, conforme à l'initial (mais exprimé différemment), voire en code source.

Dans notre travail, nous avons utilisé pour décrire nos transformations le langage ATL[7] et son moteur d'exécution. La section suivante présente les concepts essentiels des métamodèles *Mind Map*, KAOS et SYSML. Leurs descriptions complètes peuvent-être trouvées dans nos travaux précédents (Wanderley et Araujo, 2013). La description complète du métamodel SYSMLpeut être trouvée dans (Object Management Group, 2012). Un résumé des règles de transformations entre ces modèles suivent.

2.5. *Métamodèles et transformations de modèles*

2.5.1. *Le métamodèle Mind Map*

Le métamodèle présenté par (Wanderley et Araujo, 2013) décrit qu'une *Mind Map* est associée à une structure (*Structure*) et à un contenu ou un concept (*Content*). Cette structure est définie par la composition de nœuds (*Node*) et de connexions (*Edge*) formant la structure d'un graphe. Ce type de graphe peut être défini par une paire $G = (V, E)$ où $V \neq \varnothing$ est défini comme un ensemble fini d'éléments appelés sommets (ou nœuds), et *E* est une relation binaire sur *V*. L'élément *E* forme un ensemble de paires, de la forme $(vi, vj), (i, j)$ qui sont appelés bords.

Les nœuds peuvent être classifiés en groupes de nœuds (*Group*) ou en feuilles (*Leaf*), établissant ainsi une relation hiérarchique entre eux. La notation (*Notation*) est le graphe de ressources utilisé pour spécifier et différencier un nœud d'un autre dans la carte. Elle peut être représentée textuellement ou via un format XML (*Textual*), par une icône (*Icon*), cette dernière pouvant être utilisée pour tout autre élément ou par une image (*Image*). La structure d'une *Mind Map* peut également être représentée graphiquement (*GraphicalConector*), étiquetée (*Tag*) en tant que *NodeLink* ou comme une flèche *Arrow*.

2.5.2. *Le métamodèle* KAOS

La Figure 1 (Matulevicius et Heymans, 2005) montre les éléments du métamodèle KAOS essentiels à la transformation de modèles. La métaclasse *KAOS* en est la racine. Cette métaclasse est composée de nœuds et de liens (*Nodes* et *Links*). Dans le métamodèle, les nœuds peuvent être : un obstacle *Obstacle* ; un nœud opérationnel *OperationNode* ; un objet *Object* (représenté par une entitée *Entity*, dans les considérations objets, et par un agent *Agent* dans les considérations agents), et peut être un but *Goals* (traduit par un *Requirement*), une propriété de domaine (*DomainProperty*),

7. ATL - ATL Transformation Language - `http://www.eclipse.org/atl/`

une attente (*Expectation*) ou un but léger (*SoftGoal*). Un but léger est un nœud qui ne peut jamais être racine, ni feuille de type *Refinement*, d'un autre but.

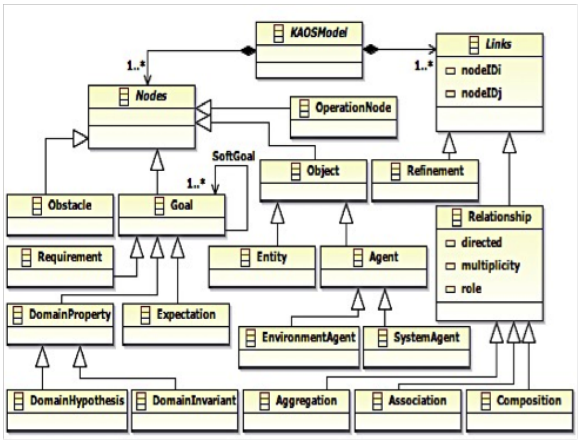


Figure 1. *Métamodèle* KAOS

Bien qu'il y ait de nombreux liens, nous nous focalisons dans cet article uniquement sur les liens de raffinement (*Refinement*) et les relations (*Relationships - Aggregation*, *Association* et *Composition*) dans le contexte du modèle objet.

2.5.3. *Le métamodèle* SysML

SysML n'a pas à proprement parlé de métamodèle. Il a été défini par un profil UML, lui-même basé sur un sous-ensemble UML appelé *UML4SysML*. Cependant, il est possible de définir un métamodèle basé sur sa définition officielle.

La Figure 2 illustre les éléments essentiels du métamodèle traitant de l'expression des exigences. Celle-ci est basée principalement sur une extension des métaclasses *Class* et *Trace* du métamodèle UML. Une exigence (*Requirement*) est une classe (*Class*) à laquelle on a adjoint deux propriétés textuelles permettant de spécifier une valeur d'identifiant (*id*) et un texte (*texte*) la décrivant. Les exigences peuvent être liées entre elles en utilisant des extensions de la métaclasse UML *Trace* : *Derive* et *Refine* permettent respectivement d'étendre et de raffiner une exigence ; *Satisfy* est utilisée afin de lier une exigence à un élément de modèle la satisfaisant ; *Verify* permet de lier un cas de test à une exigence afin de définir comment vérifier la réalisation de l'exigence dans le système. Les cas de test peuvent être spécifiés en utilisant à la fois une opération réalisant le test (structurel) et sa description comportementale.

Notre approche nécessite également un certain nombre de concepts transverses SysML, définis à partir du package *ModelElements*, tels que les points de vue («`viewpoint`») (servent à modéliser les préoccupations des utilisateurs), le lien de contenance («`containment`») (relation organisationnelle), ou encore la dépendance entre

un point de vue et une vue qui est matérialisée par la relation de contrôle («`control`»). Elle nécessite enfin la matérialisation de la notion d'acteur à travers la représentation «`actor`» définie dans le pacakge *Uses Cases*. Nous ne rentrons pas dans le détail de ces éléments de métamodèle et renvoyons à (Ahmad, 2013) pour plus de détail.

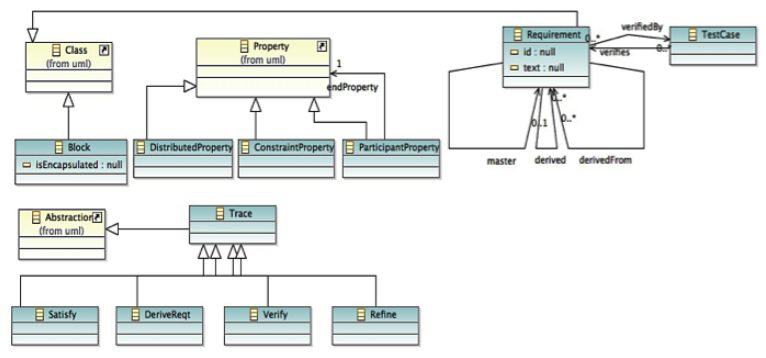


Figure 2. *Partie du métamodèle* SysML *concernant les exigences*

2.6. *Règles de transformations*

Nous décrivons ici un ensemble de règles permettant de transformer un modèle source décrit dans un métamodèle à un modèle cible décrit dans un autre métamodèle, en restant conforme au premier. Nous décrivons en premier les règles de transformations entre *Mind Map* et Kaos, puis celles entre Kaos et SysML.

2.6.1. *Transformations de Mind Map vers* Kaos

Le processus complet de transformations de Min Map vers Kaos a été décrit dans (Wanderley et Araujo, 2013) : nous y avons décrit une approche de recueil des exigences orientée modèle, permettant de générer systématiquement des modèles Kaos à partir de modèles *Mind Maps*, en utilisant un métamodèle exprimé en *Ecore*, et des transformations de modèles. Trois types de *Mind Maps* sont traitées, chacune spécifiant des préoccupations Kaos (agents, buts et objets). Après leurs transformations vers Kaos, l'ingénieur des exigences compose les modèles partiels générés afin de fournir un modèle Kaos complet. La nouveauté apportée par ce papier est de fournir un cadre permettant de passer des modèles Kaos exprimés à des modèles SysML et ainsi à fournir une série de transformations de modèles permettant de fournir un modèle des exigences SysML à partir de *Mind Maps* définies par les utilisateurs.

2.6.2. *Transformations de* Kaos *à* SysML

Nous avons identifié dans (Ahmad, 2013) un ensemble de correspondances entre des concepts Kaos et des concepts SysML. Elles sont résumées dans le tableau 1.

Concepts	KAOS	SYSML
Requirement Description	Abstract Goal, Elementary Goal	Textual Requirement
Monitoring	Contribution Goal	«satisfy»
Relationship	AND/OR, Contribution Nature (Positive, Negative), Contribution Type (Direct (Explicit), Indirect (Implicit))	«Verify», «Refine»
Dependency, Impact	Contribution Nature (Positive, Negative)	«Derive», «Contain»

Tableau 1. *Correspondance entre* KAOS *et* SYSML

Cela nous a permis de définir des règles de transformations que nous résumons ci-dessous, en les organisant en règles de transformations des exigences et règles de transformations architecturales.

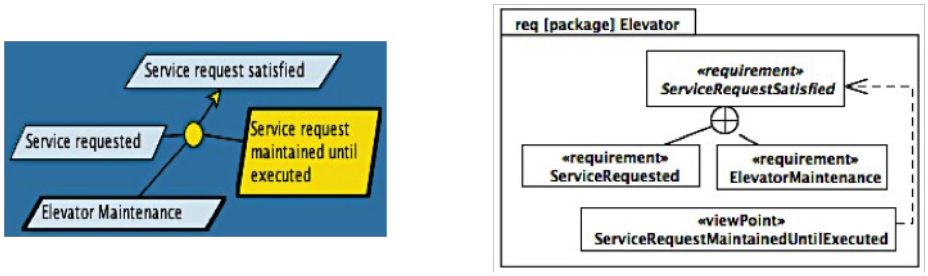


Figure 3. *Transformation d'un modèle* KAOS *vers un diagramme d'exigences* SYSML

Transformations des exigences : chaque but (*goal*) KAOS est transformé en une exigence («requirement») SYSML. Le but racine KAOS est transformé un une exigence abstraite SYSML (*abstract* «requirement»), et ses fils sont transformés en exigences SYSML à l'aide de la relation de décomposition $\otimes$. Les exigences KAOS sont vues comme des exigences systèmes SYSML et les attentes KAOS comme des exigences utilisateurs SYSML (matérialisées par les points de vue «viewPoint»). La Figure 3 illustre cette transformation.

Transformations architecturales : chaque entité KAOS (*entity*) est traduite en un bloc SYSML. Chaque opération KAOS (*operation*) est traduite en une activité SYSML ou en une opération d'un bloc. Chaque agent environnemental (*environment agent*) KAOS correspond à un acteur (*actor*) SYSML et chaque agent

système (*system agent*) KAOS à un bloc SYSML. Les Figures 4 et Figures 5 illustrent ces transformations.

Figure 4. *Objets et agents* KAOS *sources de la transformation vers* SYSML

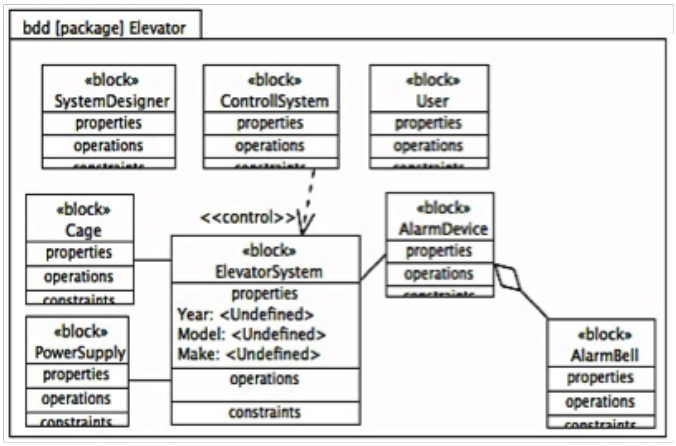


Figure 5. *Modèle de blocs* SYSML *généré*

3. Cas d'étude industriel

Cette section décrit l'application de notre approche à un cas d'étude industriel. Une partie a été présentée dans (Wanderley et Araujo, 2013) et étendue à la transformation des modèles générés vers SYSML.

Mobiciti[8] est une start-up brésilienne qui vend des produits innovants aux contenus informatifs, culturels ou publicitaires diffusables aux usagers des transports publics. *Audiobus* est un système intelligent qui utilise un système d'information géographique et diffuse les informations sous la forme d'un son numérique localisé. Des messages personnalisés et des contenus développés spécifiquement sont diffusés dans une zone donnée propre à chaque véhicule.

8. Mobciti site - `http://www.mobciti.com/`

3.1. *Génération de modèles* KAOS *pour Audiobus*

L'activité d'élicitation permettant la capture des agents, des buts et des objets a été réalisée par le client à distance via la plateforme *Hangout*. Cet environnement a été choisi pour sa facilité d'utilisation et pour ses aptitude de partage de bureau qui nous a permis de réaliser la modélisation de la *Mind Map* en ligne simultanément avec à la fois le client et l'ingénieur des exigences. Ce dernier peut poser des questions au client afin d'identifier les préoccupations principales de KAOS (i.e. agents, objets, buts). Les Figures 6 et 7 montrent le type des résultats obtenus. Notre approche définit un processus systématique permettant de générer chaque préoccupation KAOS séparément à travers une notation définie sous *Mind Map*. L'utilisation de *Mind Map* améliore la séparation des préoccupations de KAOS et aide à obtenir le modèle des buts.

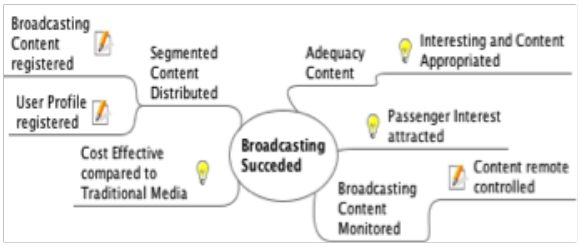


Figure 6. *Mind Map de Audiobus*

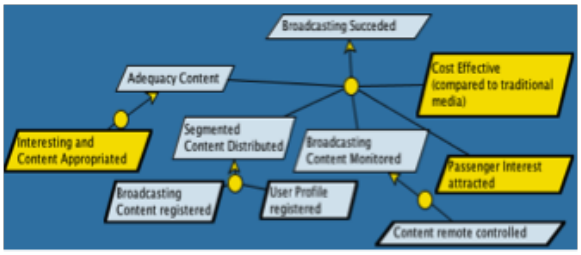


Figure 7. *Mind Map enrichie avec des annotations aidant à la transformation*

3.1.1. *Recomposition du modèle* KAOS

Après la génération des modèles KAOS partiels (i.e. modèles des buts, modèles d'agents et modèles objets), l'ingénieur des exigences se contente de les recomposer. Sa tâche consiste à assigner les responsabilités de chaque agent avec ses buts, ses exigences ou ses attentes respectifs. Il doit également raffiner le modèle objet et finalement spécifier les relations de contrôle entre un agent (composant logiciel) et les objets qu'il contrôle, ce qui donne au final le modèle KAOS décrit dans la Figure 8.

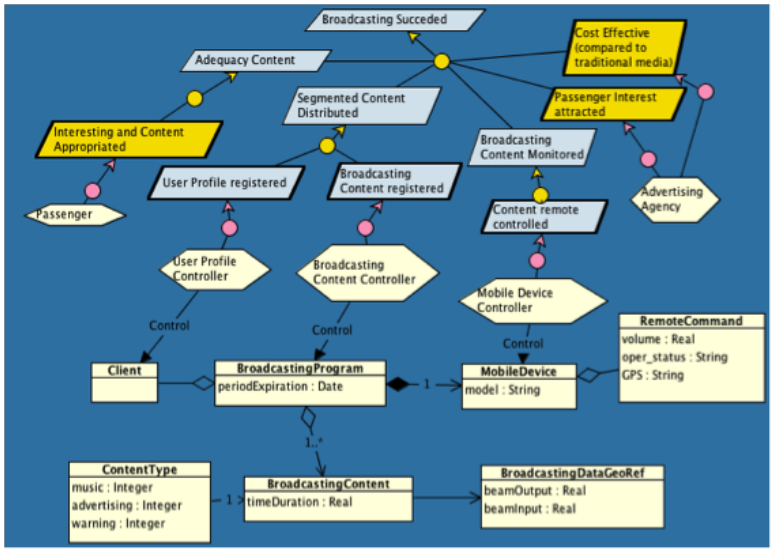


Figure 8. KAOS *du système Audiobus*

3.2. *Génération du modèle* SYSML *d'Audiobus à partir des modèles* KAOS

3.2.1. *Des buts aux exigences*

En se basant sur le modèle partiel de KAOS et sur les préoccupations générées par *Mind Map*, les buts (*Goal*), attentes (*Expectation*) et exigences (*Requirements*) d'Audiobus ont été transformés de la manière suivante (voir la Figure 9) : (i) le but racine (*BroadcastingSucceded*) a été transformé en «`requirement`» avec la propriété *abstract* positionnée à *vrai* ; (ii) les buts enfants de *BroadcastingSucceded* non définis en tant qu'attente ou qu'exigence ont été transformés en «`requirement`» et liés à leur parent (i.e. *Adequacy-Content* à l'aide de la relation de décomposition : (iii) les buts attentes (i.e. *CostEffective*) ont été transformés en point de vue («`viewPoint`») et ont été liés à leur parent à l'aide de la relation «`deriveBy`» : (iv) le but exigence (i.e. *UserProfileRegistred*) a été transformé en «`requirement`» et relié via «`satify`».

3.2.2. *Des agents et objets aux blocs*

En se basant sur les modèles partiels KAOS agents et objets générés par *Mind Map* (Figure 10), la seconde étape de notre méthode a produit un modèle en deux étapes distinctes (montrées par la Figure 11) : (i) tous les agents KAOS ont été transformés en «`actor`», après quoi l'ingénieur des exigences a dû spécifier l'acteur comme un utilisateur ou un agent système : (ii) toutes les objets (incluant les attributs et les relations) ont été transformées en «`block`» avec des propriétés (*attributes*) et les mêmes abstractions concernant les relations.

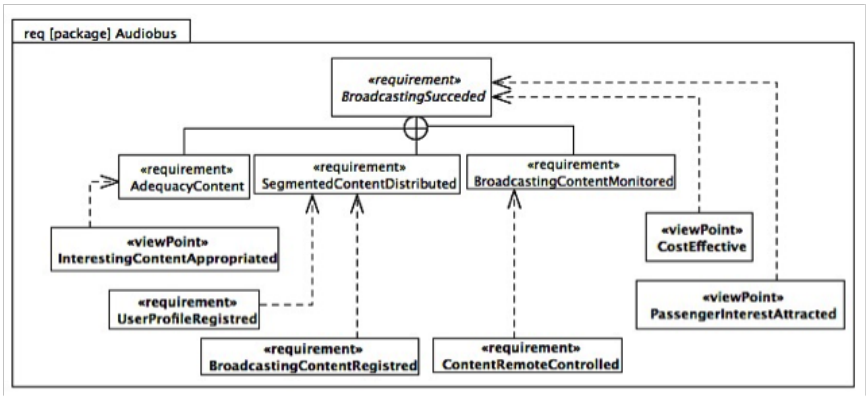


Figure 9. *Diagramme des exigences* SysML *d'Audiobus*

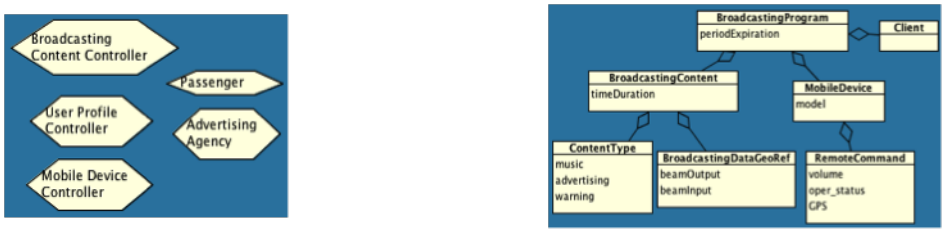


Figure 10. *Modèles* KAOS *partiels (Agents et Objets) d'Audiobus*

3.2.3. *Recomposition des modèles* SysML *d'Audiobus*

Après la transformation de chaque préoccupation de KAOS en SysML, la dernière activité de l'ingénieur des exigences a été d'annoter (manuellement) ces deux modèles SysML en les enrichissant avec des liens de traçabilité permettant d'améliorer la qualité sémantique du modèle. Dans l'exemple, chaque «`actor`» (utilisateur) est lié à son «`viewPoint`» correspondant ; chaque exigence à son «`block`» (*agent système*).

A la fin de l'étude de cas, le client s'est montré satisfait de la documentation générée (qui n'existait pas auparavant) pour son produit et a mentionné qu'elle avait été générée simplement et rapidement et avait produit un document d'exigences cohérent. Pour son prochain développement, l'entreprise envisage de mener une phase de recueil des besoins traditionnelle par une équipe et en parallèle la même étude appliquant cette approche par une autre équipe. L'objectif est d'établir une comparaison afin d'établir les apports de l'approche.

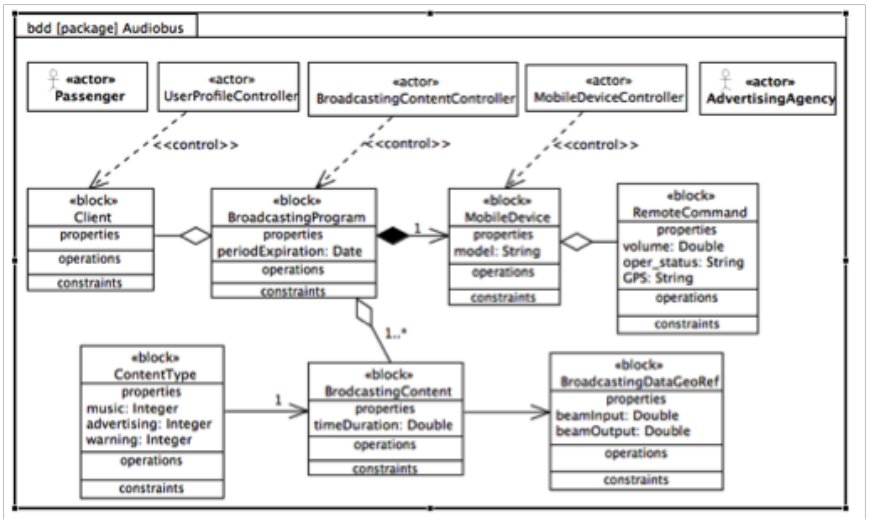


Figure 11. *Diagramme de bloc* SysML *d'Audiobus*

4. Related Works

Une étude récente (Monteiro *et al.*, 2012) a appliqué des techniques d'ingénierie des modèles aux GOREs, en réalisant une correspondance bi-directionnelle entre deux approches GORE (i* et KAOS), réalisant une plateforme interopérable qui permet de faire migrer un modèle de buts vers un autre modèle de buts via des transformations automatiques, en gardant la consistance et la traçabilité. Dans (Niu *et al.*, 2009), est proposé l'utilisation d'une plateforme permettant de tracer les aspects depuis les buts jusqu'à l'implémentation. La plateforme fournit un langage support pour modéliser les buts aspects et des mécanismes pour les transformer en programmes orientés aspects. Cette approche utilise des transformations de type modèle vers code, tandis que la notre utilise des transformations modèle vers modèle. Un état de l'art des approches d'ingénierie des modèles en rapport avec les exigences non fonctionnelles est traité par (Ameller *et al.*, 2010) : en général, elles n'y sont pas supportées. Un cadre intégrant les exigences non fonctionnelles dans le cœur du processus de transformations de modèle est décrit, mais aucune approche à part entière n'est proposée. Dans (Patricio *et al.*, 2011), un langage extensible permettant d'unifier et de représenter les langages GOREs est proposé. Le traitement unifié des concepts du modèle permet de définir plus de projections de chaque problème que les langages traditionnels. En outre des modèles hybrides peuvent être utilisés. Cependant, si l'adoption d'un langage unifié n'est pas possible, une approche basée modèle reste possible.

Pour résumer, notre approche diffère de celles présentées précédemment par le fait qu'elle utilise un modèle créatif (*Mind Map*), plus proche des experts domaine et

des utilisateurs finaux, avant de générer les modèles d'exigence, en rendant ainsi le processus de capture des exigences plus agile.

5. Conclusion

L'article définit une approche de recueil des exigences orientée modèle permettant de générer systématiquement des modèles SysML à partir de modèles KAOS, eux-même générés à partir de modèles *Mind Maps*, en utilisant les techniques de métamodélisation et de transformations de modèles. Un processus agile systématique permettant la participation active des clients a été défini et appliqué à un cas d'étude réel. La majeure contribution de ce travail est de fournir une plateforme pour faciliter la communication entre les développeurs et les experts du domaine en utilisant *Mind Map*, et de rendre plus précis, consistants et traçables les modèles utilisés, des exigences à l'analyse et la conception, par l'adoption des techniques et méthodes de l'ingénierie des modèles comme part entière de cette plateforme.

Comme travail futur, nous comptons terminer l'implémentation complète du processus de transformation en ATL, pour concevoir d'une part un protocole empirique permettant d'évaluer la compréhensibilité des *Mind Map*, et d'autre part pour inclure les *Mind Maps* comme outils permettant de capturer les opérations à inclure dans les modèles KAOS et automatiser la partie assemblage de ces modèles. Nous accompagnerons l'entreprise afin de comparer cette approche à une approche plus traditionnelle et évaluer ainsi son apport. Enfin, dans ce contexte, l'utilisation des méthodes formelles basées sur des techniques classiques de test et de preuve, devient incontournable pour certifier et valider les transformations de modèles présentées dans ce papier.

6. Bibliographie

Ahmad M., « Modeling and Verification of Functional and Non Functional Requirements of Ambient, Self-Adaptive Systems », PhD thesis, University of Toulouse, 2013.

Ameller D., Franch X., Cabot J., « Dealing with Non-Functional Requirements in Model-Driven Development », *IEEE Int. Requirements Engineering Conf.*, 2010, p. 189-198.

Antón A. I., McCracken W. M., Potts C., « Goal Decomposition and Scenario Analysis in Business Process Reengineering », *Proceedings of Advanced Information Systems Engineering*, CAiSE '94, London, UK, 1995, Springer-Verlag, p. 94–104.

Buzan T., Buzan B., *The Mind Map Book*, BBC Books, London, 2003.

Castro J., Kolp M., Mylopoulos J., « Towards Requirements-Driven Information Systems Engineering : the Tropos Project », *Information Systems*, vol. 27, n° 6, 2002, p. 365 - 389.

Dardenne A., van Lamsweerde A., Fickas S., « Goal-Directed Requirements Acquisition », *Science of Computer Programming*, vol. 20, n° 1-2, 1993, p. 3 - 50.

Davidson M. J., Dove L., Weltz J., « Mental Models and Usability », *Cognative Psychology 404*, Depaul University, novembre 1999.

Kleppe A. G., Warmer J., Bast W., *MDA Explained : The Model Driven Architecture : Practice and Promise*, Addison-Wesley, 2003.

Lamsweerde A. V., Letier E., « From Object Orientation to Goal Orientation : A Paradigm Shift for Requirements Engineering », *Workshop on Radical Innovations of Software and System Engineering*, LNCS, Springer-Verlag, 2003, p. 4–8.

Lawrence Chung Brian A. Nixon E. Y. J. M., *Non Functional Requirements in Software Engineering*, Kluwer Academic Publishers, Massachusetts, USA, 2000.

Matulevicius R., Heymans P., « Analysis of KAOS Meta-model », rapport, 2005, University of Namur, Belgium.

Mernik M., Heering J., Sloane A. M., « When and How to Develop Domain-specific Languages », *ACM Comput. Surv.*, vol. 37, n° 4, 2005, p. 316–344, ACM.

Monteiro R., Araujo J., Amaral V., Goulao M., Patricio P., « Model-Driven Development for Requirements Engineering : The Case of Goal-Oriented Approaches », *8th Int. Conf. on the Quality of Information and Communications Technology*, IEEE Computer, 2012, p. 75–84.

Moody D., Sindre G., Brasethvik T., Solvberg A., « Evaluating the Quality of Information Models : Empirical Testing of a Conceptual Model Quality Framework », *25th Int. Conf. on Software Engineering*, 2003, p. 295-305.

Niu N., Yu Y., González-Baixauli B., Ernst N., Sampaio Do Prado Leite J. C., Mylopoulos J., « Transactions on Aspect-Oriented Software Development », chapitre Aspects Across Software Life Cycle : A Goal-Driven Approach, p. 83–110, Springer-Verlag, 2009.

Object Management Group, « OMG Systems Modeling Language V1.3 », rapport, 2012, Object Management Group.

Patricio P., Amaral V., Araujo J., Monteiro R., « Towards a Unified Goal-Oriented Language », *35th IEEE COMPSAC*, 2011, p. 596–601.

Pressman R. S., *Software Engineering : A Practitioner's Approach*, McGraw Hill, 2006.

Riviére A., Benac C., « MBSE using SysML :A350 XWB Experience », 2010.

Rolland C., Salinesi C., « Modeling Goals and Reasoning with Them », *Engineering and Managing Software Requirements*, p. 189-217, Springer, 2005.

Schmidt D., « Guest Editor's Introduction : Model-Driven Engineering », *Computer*, vol. 39, n° 2, 2006, p. 25-31.

Sommerville I., *Software Engineering*, Addison-Wesley, 9th édition, 2010.

University of Toronto, « GRL - Goal-oriented Requirement Language », Toronto, Canada.

van Lamsweerde A., « Goal-Oriented Requirements Engineering : a Guided Tour », *5th IEEE Int. Symp. on Requirements Engineering*, 2001, p. 249-262.

Wanderley F., Da Silveira D. S., Araujo J., Lencastre M., « Generating Feature Model from Creative Requirements Using Model Driven Design », *16th Int. Software Product Line Conference - Volume 2*, ACM, 2012, p. 18–25.

Wanderley F., Araujo J., « Generating Goal-Oriented Models from Creative Requirements using Model-Driven Engineering », *Int. Work. on Model-Driven Requirements Engineering*, 2013, p. 1-9.

Yu E., « Modelling Strategic Relationships for Process Reengineering », PhD thesis, Graduate Depart-ment of Computer Science, University of Toronto, 1995.

Session 3

Processus : traces, fouilles et modélisation

Modélisations dans une approche de veille générique (GWatch) :

Clustering centré acteurs de veille

Mseddi Rim[1], Sahbi Sidhom[2], Malek Ghenima[3] et Henda Ben ghezela[4]

1. *RIADI-GDL-Manouba, Email : b_mseddirim@yahoo.fr*
2. *LORIA-Université de lorraine, France, Email : sahbi.sidhom@loria.fr*
3. *ESCEM-Manouba, Email : malek.ghenima@esem.rnu.tn*
4. *RIADI-GDL-Manouba, Email: henda.hg@cck.rnu.tn*

RESUME. Dans le cadre de notre recherche sur la modélisation de l'acteur de veille et le développement d'un système de veille générique centré sur l'acteur, nous présentons dans cet article une nouvelle approche d'intégration semi-automatique d'un processus de veille associé à un système de recherche d'information existant. Cette approche vise essentiellement à faciliter la mise en place d'un système de veille, ce qui permettra, d'un côté, la définition des besoins de veille à partir des informations collectées et, d'un autre côté, la structuration puis l'évolution des projets de veille dans une approche systémique avec l'information traitée.

ABSTRACT. As part of our research on watch actor modeling and the development of a generic watch system centered on the actor, we present in this paper a new semi-automatic integration approach to watch process with existing information retrieval system. This approach is essentially to facilitate the establishment of a watch system associated, which will, on the one hand, the definition of the watch needs from information collected and, on the other hand, the structure and the evolution of intelligence projects in a systemic approach with processed information.

MOTS-CLES : système de veille, recherche d'information, gestion de la connaissance, intégration, modélisation de l'acteur, clustering.

KEYWORDS: Watch system, information research, Knowledge organisation, integration, actor modeling, clustering.

1. Introduction

Si la veille a maintenant acquis ses lettres de noblesse, elle reste, comme tout concept récent, encore mal comprise » affirme J.P. Bernat (Bernat, 2008). Il existe plusieurs définitions de la veille qui peuvent concerner soit les ressources, soit les outils ou les types de veilles utilisés. Le plus souvent, elle est considérée comme une activité, démarche ou ensemble de méthodes et de techniques de gestion de l'information à valeur ajoutée (définition des besoins, recherche, collecte, traitement et diffusion). On distingue aussi entre la veille qui utilise des ressources et des outils hypertextuels, web natif, web 2.0 ou web conceptuel (Delaby, 2004; Leitzelman, 2009a; Pateryon, 1998; Raquel, 2011).

Dans un processus de veille, la première étape à savoir la définition des besoins de veille reste la tâche la plus difficile pour un acteur de veille d'autant lorsque ces besoins sont imprécis ou lorsque le corpus ne lui est pas familier. Dans notre contexte, l'objectif est de développer le système de veille afin de répondre le plus pertinemment possible. Pour la veille, nous considérons trois types d'information comme ressources au processus: le jugement de la pertinence par l'acteur de veille, des clusters récupérés à partir du corpus initial retournés par le système de recherche d'information et des données globales introduites explicitement par les différents acteurs.

Le premier type repose sur le retour de pertinence explicite de l'acteur de veille à travers la sélection des documents retenus dans des projets de veilles antérieurs. Le deuxième type repose sur l'analyse du corpus afin d'établir des relations entre les clusters et les environnements de veille de l'acteur. Pour finir, le dernier type encapsule toutes les informations explicites qui alimentent le système de veille. Elles peuvent parvenir de l'animateur, du veilleur ou même de l'utilisateur final à travers la sélection de ces centres d'intérêt, des environnements de veille, etc.

Dans cet article, nous présentons, dans le premier paragraphe, la gestion des informations dans un processus de veille. Dans un deuxième paragraphe, nous décrivons notre approche et nous détaillons, dans le dernier paragraphe, nos résultats d'analyse. Nous finirons par une conclusion et la présentation de quelques perspectives.

2. La gestion des informations dans un processus de veille

Les outils de veille intégrés proposés sur le marché diffèrent selon les ressources gérées et les outils de traitement de l'information proposés. En général, ils proposent la même démarche qui débute par la définition des besoins sous forme de requêtes qui peuvent être soit ponctuelle (projet), cyclique (rapport) ou continue (alerte). Ces requêtes peuvent concernées des sources spécifiques ou un scannage général du web. Les informations retournées sont sous forme de flux RSS, de pages web, de Newsletters…. Certain outils de veille proposent aussi des outils de traduction ou de génération de résumé automatique.

Les informations collectées seront, ensuite, traitées par le veilleur qui va sélectionner l'information pertinente, nettoyer les résultats des données superflues, générer l'information à valeur ajoutée et la partager en diffusion.

Dans la littérature, plusieurs processus de veille ont été proposés, nous pouvons citer les travaux de Lesca et al. qui insiste sur l'importance de la création collective de sens et la mémorisation des connaissances tacites (LESCANNING, VAS-IC) (Lesca, 2011). D'un autre côté, la méthode EQUA²te proposée par Thierry O. et Amos D. (Thiery, 2002) montre l'apport de l'annotation sur des solutions antérieures afin de les associées aux problèmes de recherche d'information actuels. Dans le même contexte, Sidhom propose le modèle SIMBAD (Sidhom, 2002) qui montre, en amont et en aval d'un cycle de veille, l'apport de l'annotation, d'une part, dans la clarification des objectifs en formalisant les indicateurs de recherche d'informations, et d'autre part, la valorisation des informations à valeur ajoutée dans la prise de décision ou l'identification des signaux faibles.

L'aspect collaboratif du processus de veille est très important car à travers des interactions dynamiques qu'on peut assurer une pratique évolutive et constructive des connaissances collectives (Fayard, 2002). Dans cette réflexion, plusieurs méthodes ont été mises en oeuvre afin de faciliter cette dynamique de groupe. Nous pouvons citer la méthode PUZZLE (Raquel, 2011) qui consiste à faire agir les différents acteurs en leur proposant des brèves d'informations. Jakobiak propose aussi une méthode collective pour l'innovation à travers sa théorie des ingrédients (Jakobiak, 2005).

D'autres méthodes se sont focalisées sur la recherche d'information anticipative par l'identification (ou la détection) des signes d'alertes précoces. Ces informations peuvent indiquer un changement possible dans l'environnement surveillé. Dans cette approche, la veille est observée stratégique et alimentera une base de connaissances actionnables pour la gestion des risques et des opportunités de l'organisation (Lesca, 2001; Sidhom, 2011). Quant aux retours d'expérience, la pratique de veille dans les organisations et les entreprises montrent des défaillances majeures dans le bon fonctionnement de ce processus, à savoir: le manque d'implication du personnel, la surcharge cognitive, la perte dans l'hyperespace, le manque de pertinence des résultats retourné, etc.

Dans ce qui suit, nous allons développer trois types essentiels d'information qui sont gérés par un système de veille, à savoir : le profil veilleur, le profil document et le profil requête.

2.1. Le profil veilleur

Les études sur le profil veilleur que nous avons examiné dans la littérature grise sont peu nombreuses ce qui peut être expliqué par une focalisation sur les pratiques et les finalités de veille. On oublie que sur cet acteur, repose l'intégralité du processus de veille. Pour définir un veilleur, Kislin rapproche le veilleur à l'analyste en intelligence économique. Il le définit comme « *un décideur particulier qui interagit à la fois dans le domaine de l'information et avec son homologue dans le domaine économique* » (Kislin, 2007). D'un autre côté, Thomas le présente comme

un «*travailleur du savoir qui est capable de surveiller ce qui l'intéresse quotidiennement*» (Thomas, 2008).

Si nous tentons de comparer le profil d'un veilleur et le profil d'un utilisateur d'un système de recherche d'information (SRI), d'une manière générale, nous pourrons déduire qu'un veilleur partage quelques caractéristiques avec cet utilisateur (informations générale ou spécifique, centres d'intérêt, communautés,…), mais présente des spécificités qui dépendent directement de :

- Son rôle dans le processus de veille : un veilleur peut être un animateur, veilleur ou utilisateur final d'un projet de veille. Ce rôle peut changer d'un projet à un autre. Un veilleur va, donc, avoir des droits d'accès, de consultation (à des dossiers sensibles) et de traitement de l'information qui diffèrent d'un projet à un autre.

- La sélection des informations pertinentes : dans un SRI des méthodes ont été établies afin de calculer la pertinence d'une source par rapport à un utilisateur comme le calcul de nombre de clique, le temps passé sur un lien … Alors que pour un système de veille, le veilleur sélectionne les informations pertinentes ce qui offre une réponse claire au système sur les centres d'intérêt du veilleur.

- Le travail en réseau : La communauté de veille permet de définir les tendances globales d'un veilleur comme ses axes de recherche, sa position dans la communauté …Ce profilage permet alors de construire d'une manière collaborative du sens et offrir une plus-value à l'information traitée (David, 2006).

2.2. Le profil document

Le profil document permet de présenter les aspects génériques ou spécifiques des informations qui peuvent intéresser un veilleur ou une communauté de veille. Il prend en considération trois aspects :

- Les sources d'information : web, flux RSS, base de données… Dans un système de veille, le veilleur ou l'animateur de veille sélectionne les sources d'information qui sont jugées intéressants et fiables pour un ou plusieurs projets de veille. Ces sources peuvent être formelles (base de brevets, appels d'offre, documents officiels…) ou informelles (information de terrain, forum, émission radio….). On peut aussi distinguer entre les ressources web2.0 (flux RSS, forum, web social, wiki,…) et les ressources du web natif (site web, email, …) (Bourdier, 2007).

- Le support de l'information: texte, image, vidéo, son…, la gestion de l'information peut varier selon le support sur lequel elle se présente. La prise en considération des annotations, des tags ou des commentaires qui accompagne les objets multimédias et le traitement de texte comme le nettoyage des données superflus, le résumé automatique ou la traduction sont des fonctionnalités offertes généralement par les outils de veille intégrés.

- La thématique de l'information : choisir les thèmes, catégories ou sous catégories susceptibles d'intéresser le veilleur ou la communauté de veille. La

sélection de ces centres intérêts permettra de créer des alertes afin d'avertir les veilleurs sur des nouvelles données qui répondent à leurs problématiques. A ce niveau, il s'agit d'une diffusion, ciblée et automatique des informations dans les réseaux de veille.

Les profils documents sont alimentés au début d'un projet de veille par l'animateur de veille ou le veilleur. Ils seront, ensuite, enrichis d'une manière semi-automatique au cours du cycle du projet par les collaborateurs (communauté de veilleurs).

2.3. Le profil requête

Dans un projet de veille, la définition des requêtes reste l'étape la plus difficile à réaliser par le veilleur, l'animateur ou la communauté de veille. Elle nécessite une connaissance assez élaborée des besoins en veille. Plusieurs auteurs ont proposé des outils d'aide à la formulation de la requête en prenant en considération les centres d'intérêts, les variables booléennes qui permettent d'intégrer un ou plusieurs champs dans la recherche et l'aspect utilisateur (Harbaoui, 2009; Leitzelman, 2009b). Néanmoins, la difficulté reste liée à la compétence du veilleur qui doit être informé en continue de tout changement dans les environnements de l'entreprise. Il doit aussi être à l'écoute de tous ce qui l'entoure (prévoir des réunions de groupes, se déplacer dans les différents services, …) afin d'éviter le syndrome du veilleur isolé.

Dans le paragraphe suivant, nous présentons notre processus d'intégration d'un système de veille générique qui regroupera ces différents profils dans un objectif de réutilisation optimale.

3. Processus d'intégration d'un processus de veille

Dans ce qui suit, nous allons détailler notre approche d'intégration d'un processus de veille générique à un système de recherche d'information existant en réutilisant les résultats de ce dernier (fichier direct, fichier inverse, noyau d'index…). Notre processus est centré acteur dont l'objectif est d'offrir des informations jugées pertinentes à des utilisateurs cibles. Il est composé de trois phases à savoir : l'intégration, la décomposition et la recomposition de l'information.

3.1. Intégration

Notre point de départ est un système de recherche d'information conçu avec la plateforme open source « Solr » sur un corpus de données bibliographiques sur les nanotechnologies contenant 2893 références bibliographiques complètes regroupant plusieurs domaines de recherche. Nous obtenons ainsi notre matrice terme-document (Noyau0) comme entrée à notre processus de veille.

- La description de la matrice A terme-document :

	I1	*I2*	*I3*	*...*	*In*
D1	0.29	0.00	0.00	…	0.29
D2	0.00	0.00	0.37	…	0.27
...	…	…	…	…	…
Dm	0.00	0.39	0.47	…	0.25

Avec des occurrences en items (I)

I1: Photonic	***D1***: id---215
I2: Crystal	***D2***: id---306
I3: Photonic Crystal	***Dm***: id---1417
In: Photon	

A la suite, nous procédons à la décomposition en valeurs singulières de cette première matrice, afin d'obtenir des clusters représentatifs des différents environnements couverts par ce corpus.

- La décomposition singulière de la matrice A ($A= U\Sigma V^T$) :

$S_{i,p}$ = terme $_i^T$ *terme $_p$ $\qquad$ $Z_{j,q}$ = document $_j^T$ * document $_q$

0.08	0	0	0.08
0	0.15	0.18	0.10
0	0.18	0.35	0.21
0.08	0.10	0.21	0.21

0.14	0.07	0.08
0.07	0.2	0.25
0.08	0.25	0.3

La matrice **U** contient les vecteurs propres de S et la matrice $\mathbf{V^T}$ contient les vecteurs propres de Z.

$$\overset{\frown}{\text{Terme}}{}_i^T \rightarrow \{[U_1]\dots [U_L]\}. \begin{Bmatrix} \sigma_1 \dots 0 \\ 0 \dots \sigma_L \end{Bmatrix} \begin{Bmatrix} [V1] \\ . \\ . \\ [V_L] \end{Bmatrix}$$

Les valeurs singulières ($\sigma_{1\dots}\ \sigma_L$) peuvent alors être sélectionnées à une approximation K afin d'obtenir des clusters à précisions variables et ajustées.

Cette première étape nous a permis d'obtenir les résultats suivants sous forme d'un fichier XML (cf fig1 et tab1).

```
▼<arr name="labels">
    <str>Distortion in Microwave Photonic Filters</str>
  </arr>
  <double name="score">4.162376843146952</double>
▼<arr name="docs">
    <str>1002</str>
    <str>1030</str>
  </arr>
 </lst>
▼<lst>
 ▼<arr name="labels">
    <str>InAs GaAs Quantum Dots</str>
  </arr>
  <double name="score">20.144567714647327</double>
 ▼<arr name="docs">
    <str>211</str>
    <str>347</str>
  </arr>
 </lst>
▼<lst>
 ▼<arr name="labels">
    <str>One-way Quantum</str>
  </arr>
  <double name="score">32.09865026310469</double>
 ▼<arr name="docs">
    <str>1052</str>
    <str>1238</str>
  </arr>
 </lst>
▼<lst>
```


Figure 1. Fichier XML contenant les différents clusters

Tableau 1. Échantillon de clusters extraits

Cluster	Score	Nbre de doc
Silicon	8,481	96
Resonators	1,974	96
Quantum	6,865	88
Applications	6,455	78
Silicon Photonic	5,139	46
Liquid Crystal	40,5	34
Supercontinuum Generation	21,238	32
Quantum Dots	26,181	33
Photonic Crystal Resonant	5,807	33
Photonic Quantum	14,396	34
Fiber Sensor	2,626	27
Laser Pulses	29,073	18

Pour un ensemble de K traité, nous avons obtenues des résultats de clustering pertinents pour un k=2, générant en totalité 89 clusters. Nous avons créé un nouveau noyau (Noyau 1) sur la plateforme Solr et dans l'objectif est de synthétiser de plus

en plus nos environnements. Nous avons regroupé les clusters obtenus dans des Meta-clusters qui regroupent deux ou plusieurs clusters. Nous avons placé les clusters orphelins dans la catégorie « OtherTopics ».

Figure 2. Fichier Json contenant les Meta-Clusters

Le résultat obtenu est sous forme d'un fichier Json contenant les Meta-Clusters. Les clusters regroupent ainsi les identifiants des documents qu'ils représentent. Ces résultats seront enregistrés dans un troisième noyau (Noyau 2) Solr afin de faciliter la recherche dans ces Meta-Clusters.

La deuxième phase de notre processus de veille générique présente la décomposition de l'information.

3.2. Décomposition

L'objectif de cette deuxième phase est de décomposer les environnements de veille sur les différents veilleurs (ou communautés de veille) selon leurs centres d'intérêt. Afin de tester notre application nous avons construit quatre profils de veilleur qui représentent des environnements différents.

- Profil requête : à la différence d'un système de recherche d'information, le veilleur va sélectionner les documents qu'il va juger pertinent. La matrice requêtes/documents est alors une matrice binaire (1 pour les documents sélectionnés et 0 pour les documents ignorés)

$$\begin{array}{c} \\ D_1 \\ D_2 \\ \dots \\ D_m \end{array} \begin{array}{c} \begin{array}{ccc} Rq_1 & Rq_2 \dots & Rq_n \end{array} \\ \begin{pmatrix} 1 & 0 & 1 \\ 0 & 0 & 0 \\ \dots & & \\ 0 & 1 & 1 \end{pmatrix} \end{array}$$

- Centres d'intérêts : le veilleur sélectionne ses centres d'intérêts en amont du processus de veille. La matrice centre d'intérêt/veilleur est aussi une matrice binaire (1 pour les centres d'intérêts sélectionnés et 0 pour ceux ignorés).

	CI_1	CI_2	CI_x
V_1	1	0	0
V_2	0	1	1
…	…		
V_k	1	1	0

- Profil document : le profil document permet de spécifier les préférences du veilleur concernant les résultats retournés.

La première étape est la classification de tous ces champs représentatifs du veilleur dans une base de connaissances spécifique à chaque veilleur (cf. Fig3).

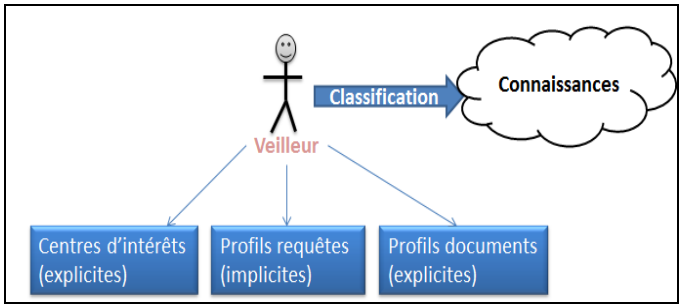


Figure 3. Projection du veilleur sur l'axe des connaissances

La deuxième étape consiste en un « matching » supervisé des différents veilleurs par rapport au Meta-clusters extraits dans la première phase du processus (intégration) : (cf. Fig4)

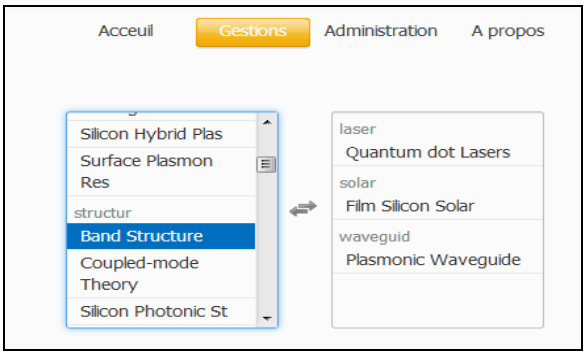


Figure 4. Interface sélection des Meta-Clusters

Le résultat de cette deuxième phase est une matrice de décomposition Decomp{MetaClust}.{Veilleur}. La dernière phase la recomposition sera présentée dans le paragraphe suivant.

3.3. Recomposition

La décomposition des informations selon les préférences implicites et explicites de chaque veilleur a permis de mieux viser les besoins de ce dernier afin de remédier au problème majeur des systèmes de veille qu'est la surcharge cognitive. Néanmoins, le résultat final attendu par le décideur ou la communauté d'experts doit regrouper toutes les briques de résultats retournés par les veilleurs. Cette dernière phase consiste alors à regrouper les résultats qui sont présentés sous forme de documents sélectionnés, de les présenter sous forme d'informations à valeur ajoutée et de les diffuser dans les réseaux concernés. Chaque veilleur sélectionne les articles qu'il juge pertinents (cf Fig.5). Ensuite, il les diffuse sur le réseau.

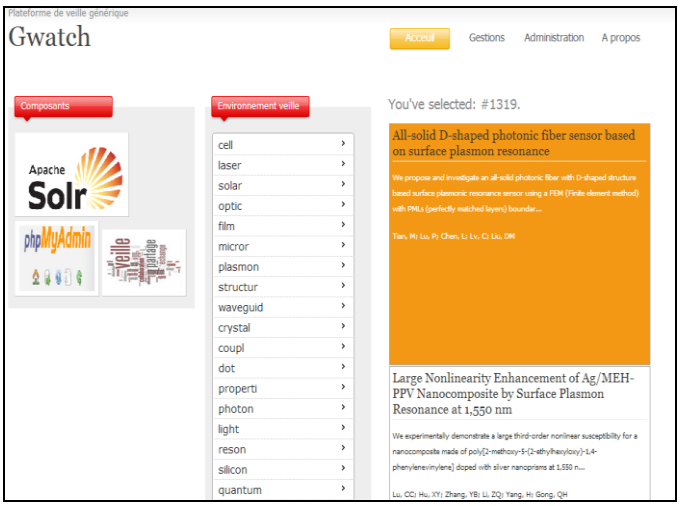


Figure 5. Plateforme principale Gwatch

Le regroupement des résultats obtenus est une matrice binaire document/ veilleur où on attribue la valeur 1 à chaque document i sélectionné par un veilleur j et 0 sinon.

$$\begin{array}{c} \\ V1 \\ V2 \\ \dots \\ Vk \end{array} \begin{pmatrix} D1 & D2 & Dm \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ \dots & & \\ 0 & 0 & 1 \end{pmatrix}$$

La hiérarchisation des informations a permis de prendre en considération deux aspects : la pertinence de l'information et la position du veilleur dans le réseau de veille. En illustration sur un exemple :

Veilleur	H
V1	1
V2	2
V3	2
V4	3

Document	H
D1	1
D2	2
D3	3

Le processus de veille proposera aux veilleurs, les résultats suivants:

V1{D1,D2,D3}; V2{D1,D2}; V3{D2,D3}; V4{D1}

4. Résultats et discussion

Un système de veille, étant un système d'information (SI), sa mise en place comporte comme tout SI quatre phases successives à savoir : l'étude, le développement, l'implémentation et la clôture(Caron-Fasan, 2010; Schwalbe, 2004). La phase clôture ou bilan permet d'évaluer l'apport d'un tel système ainsi que ses limites.

Notre système de veille générique, dans son état actuel, est un prototype complexe qui a été simplifié, qui vise essentiellement à faciliter la mise en place d'un système de veille tout en prenant en considération les besoins spécifiques du veilleur et les différentes facettes d'un projet de veille. Notre prototype ne contient pas toutes les fonctionnalités d'un système de veille intégré classique (résumé automatique, traduction, gestion des flux RSS…). Néanmoins, il apporte une réponse claire aux deux problèmes majeurs des systèmes sur le marché, à savoir la résolution de la surcharge cognitive et la perte dans l'hyperespace. La centralisation sur le profil acteur, ses besoins, ses projets antérieurs et sa position dans la communauté de veille, nous ont permis d'obtenir des interfaces spécifiques pour chaque acteur et de viser l'information cible selon ses besoins.

La construction de notre système de recherche d'information par la plateforme Solr, sur le corpus de données bibliographiques en nanotechnologie, nous a permis d'obtenir, comme entrée à notre système de veille générique, le fichier direct et le fichier inverse (CF Fig.6) correspondant au corpus d'analyse.

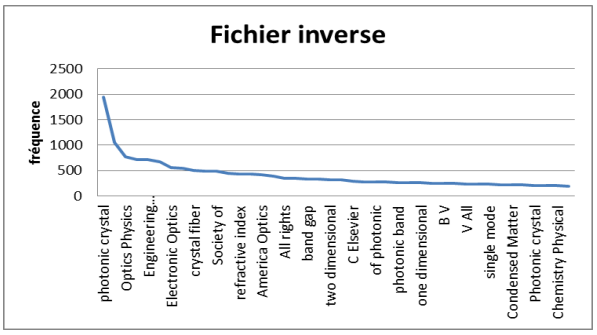


Figure 6. Fichier inverse index/fréquence (loi de Zipf)

Nous avons construit grâce à la méthode de décomposition en valeurs singulières par un algorithme de clustering non supervisé Lingo qui est implémenté dans la plateforme Solr par l'API carrot2[21]. Nous avons obtenu pour la valeur arbitraire K=2 le nombre de 89 clusters (voir figure suivante) avec un degré de pertinence. Un nombre de documents n'ont pas été classés et ils ont été placés dans un cluster général (divers) (cf. Fig7).

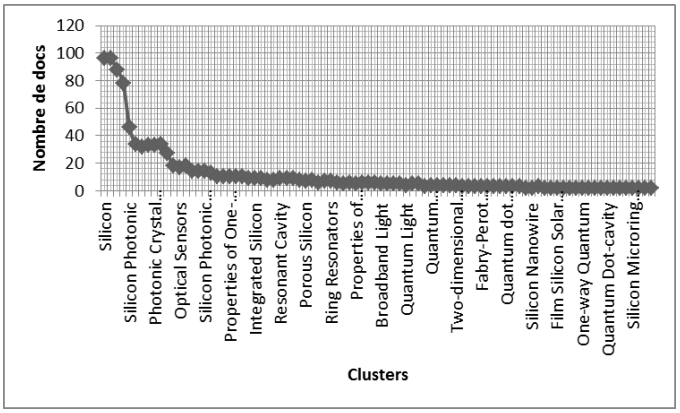


Figure 7. Courbe des clusters/nombres de documents

Dans l'objectif de synthétiser nos résultats, nous avons regroupé ses différents clusters avec un algorithme de clustering non supervisé dans des méta-clusters. La figure 8 schématise le nombre de cluster par Meta-cluster tout en sachant qu'un méta-cluster regroupe au moins 2 clusters et qu'un cluster peut se trouver dans un ou plusieurs Meta-clusters.

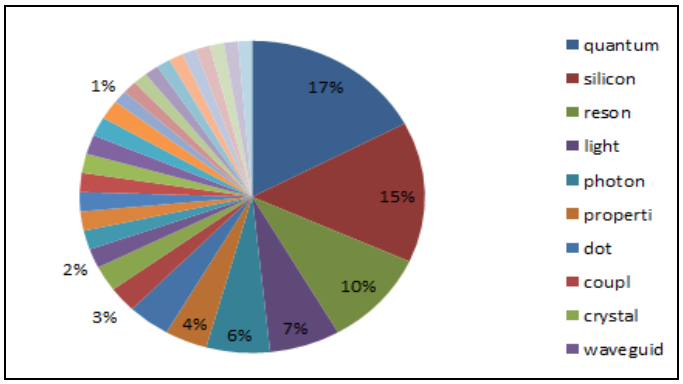


Figure 8. Classification des clusters dans des Meta-clusters

Dans le contexte de veille, l'évaluation du système proposé ne répond pas aux mêmes critères (calcul de précision et de rappel..) qu'un système de recherche d'information (Baccini, 2008) même si ce dernier est considéré comme un noyau pour un système de veille et donc sa pertinence est très importante pour le retour d'informations recherchées. Des évaluations ont été proposées, comme à titre d'exemple, les phases de veille couverts, les outils et fonctionnalités offerts, les sources et types d'informations gérés… D'un autre côté, on peut proposer des évaluations sur l'impact d'un tel projet dans une entreprise ou une organisation: l'implication des personnels, la pertinence des retours, l'impact sur le processus décisionnel … (Jakobiak, 2002).

A cet effet, nous proposons un tableau montrant nos apports par rapport aux différentes phases d'un projet de veille à savoir : définition des besoins, recherche, collecte, traitement et diffusion (cf Tab.2)

.

Tableau 2. Les apports de notre approche

Phase	Apports
Définition des besoins	Faciliter la définition des besoins grâce à un palmarès de Meta-clusters et clusters afin de mieux se situer par rapport au corpus.
Recherche	Précision dans la recherche par la sélection préalable de la plage des documents susceptibles d'être pertinent pour le veilleur.
Collecte	La sélection directe des documents jugés pertinents par le veilleur et la constitution d'un panier de documents qui répond à la problématique de veille.
Traitement	La possibilité de nettoyer les documents des informations jugées superflues.

Phase	Apports
Diffusion	Le regroupement et la hiérarchisation des résultats permettent d'obtenir un retour plus pertinent pour l'utilisateur final (décideur, expert…). La sécurité de l'information est assurée par la sélection au préalable par le veilleur ou l'administrateur de la communauté qui peut accéder à ces résultats.

Les limites de ce travail de recherche restent dans la possibilité de développer un système de veille complet ce qui nécessite un travail d'équipe de développeurs et plusieurs mois de développement. Néanmoins, nous considérons que notre prototype est assez représentatif de notre approche. Pour compléter notre phase test avec l'aspect utilisation, nous avons besoin d'une validation par une équipe de veilleurs spécialisées en Nanotechnologie, dans plusieurs catégories et la validation de plusieurs projets distincts.

Nous terminons cet article par notre conclusion et la présentation de nos perspectives.

5. Conclusion et perspectives

Le contexte de la recherche, à savoir les systèmes de veille, nous a permis d'identifier, d'un côté, que le noyau de tout système de veille est un SRI qui exécute des requêtes et permet l'indexation des informations collectées, et d'un autre côté, la place centrale du veilleur dans le système. Ces constats nous ont permis de nous orienter vers un système de veille générique centrée acteur et de coupler un système de recherche d'information existant, pour réutiliser ses résultats (fichier directe et inverse) et ses composants (noyau d'indexation).

Notre proposition a porté sur une approche comprenant trois phases, à savoir :

- l'intégration des informations par un processus non supervisé de clustering et l'identification des Meta-clusters ;
- La décomposition des informations selon les besoins de chaque veilleur ;
- La recomposition des résultats dont l'objectif est de fournir une réponse complète à la problématique de veille.

L'application de cette approche sur un corpus de données bibliographiques sur la Nanotechnologie comprenant 2893 références, nous a permis d'identifier 89 clusters répartis sur 34 Meta-Clusters. Nous avons ensuite décomposé un nombre de ces Meta-clusters sur quatre profils veilleur. Notre prototype « GWatch » développé a permis alors de répondre à des besoins spécifiques en offrant des informations collectées à des veilleurs ciblés.

Dans nos perspectives, nos futures orientations consistent essentiellement à pouvoir enrichir notre prototype par les différentes fonctionnalités qu'offre un système de veille intégré (Gestion des flux RSS, résumé automatique, traduction, surbrillance des mots clés dans les articles…) et sa validation par une équipe d'experts dans le domaine de la Nanotechnologie.

Bibliographie

Baccini, A., Déjean,S. ,Kompaoré , D., Mothe, J. (2008). Analyse des critères d'évaluation des systèmes de recherche d'information. *ISI_BDKM*.

Bernat, J.-P., al. . (2008). Les contours de la veille *Documentaliste-Sciences de l'Information* 45. 32-44

Bourdier, S. (2007). Enjeux et apports du web 2.0 pour la circulation de l'information dans l'entreprise: le cas du service de veille stratégique du groupe Yves Rocher. *Institut national des techniques de la documentation*.

Caron-Fasan, M.L., Lesca,H. (2010). Facteurs de risque de la conduite de projet de mise en place d'un dispositif de veille anticipative dans plusieurs Caisses d'Allocations Familiales *Colloque de l'AIM, La Rochelle*. 20 pages

David, A. (2006). La recherche collaborative d'information dans un contexte d'intelligence économique. *Le système d'information de l'entreprise*.

Delaby, A. (2004) "Le concept de veille", UFR Tours -Ecole Doctorale

Fayard, P.M. (2002). "Le concept de "Ba" dans la voie japonaise de la création du savoir". *Ambassade de france à Tokyo : Service pour la science et la technologie* SMM03-046.

Harbaoui, A., M. Ghenima, S. Sidhom. (2009). Enrechissement des contenus par la réindexation des usagers: un état de l'art sur la problématique. *SIIE*.

Jakobiak, F. (2002). Evaluation de la veille technologique. *Intelligencia competitiva*.

Jakobiak, F. (2005). *De l'idée au produit: Veille-R&D-Marché*. Editions d'organisatuin

Kislin, Phil. (2007) "Modélisation du problème informationnel du veilleur dans la démarche d'intelligence économique" in *Sciences de L'Information et de la Communication*. Nancy, Université Nancy 2

Leitzelman, M. (2009a). Etat de l'art et tendances sur le marché de la veille et l'intelligence compétitive. *ISICIL*.

Leitzelman, M., Ereteo,G., Grohan, P., Herledan, F., Gandon, F., Buffa ,M. (2009b). De l'utilité d'un outil de veille d'entreprise de seconde génération. *IC*.

Lesca, H. . (2001). Veille stratégique : passage de la notion de signal faible à la notion de signe d'alerte précoce. *VSST*.

Lesca, N. (2011). Etat des lieux des pratiques de « veille logistique durable » : une approche qualitative, Rapport. *ANNEXE N°3 au Rapport scientifique d'étape VLD.1-PREDIT 4-ADEME*.

Pateryon, E. (1998). La veille stratégique. *Economica.*

Raquel, J.M., Fabio D.B, Lesca H., Freitas H. (2011). Application de la veille anticipative stratégique pour le suivi de l'environnement et la production de connaissances actionables *Journal of Information Systems and Technology Managemet* 8. 425-440

Schwalbe, K. (2004). *Information technology project management*. 3 ed.

Sidhom, S. (2002) "Plate-forme d'analyse morpho-syntaxique pour l'indexation automatique et la recherche d'information : de l'écrit vers la gestion des connaissances". France, Université Claude Bernard - Lyon 1

Sidhom, S., P. Lambert. (2011). Information Design for weak signals detection in Economic intelligence. *SIIE*. 123

Thiery, Odile, David, Amos. (2002). Modélisation de l'utilisateur, Systèmes d'Informations Stratégiques et Intelligence Economique. *Revue Association pour le Développement du Logiciel (ADELI)* N 47. 12 p

Thomas, A. (2008). Parce que la veille bouge. *Documentaliste-Sciences de l'Information* 45. p.30-31

Adnosco: trace user data for the user

Nadia Bennani* — **Fabien Duchateau**** — **Előd Egyed-Zsigmond*** — **Philippe Lamarre***

* *Université de Lyon*
CNRS, INSA-Lyon, LIRIS, UMR5205, F-69621, France
** *Université de Lyon*
CNRS, Université Lyon 1, LIRIS, UMR5205, F-69622 France
firstname.lastname@liris.cnrs.fr

ABSTRACT.
The development of the web has seen the explosion of web forms as a mean for the user to provide information to web applications in the context of both personal and professional activities. Currently, user personal data are stored and managed at companies' side. That makes the user dependent on the corresponding services while granting her a very passive role. We aim to demonstrate that if correctly modeled, user traces provide many benefits for both the user and services. In this paper, we propose a trace model including a new web form qualification model and a user-friendly solution that improves web users productivity by providing semantics-based tools. One of our goals is to provide trace analysis methods that enable answering questions like : "Where have I read something about Inforsid on the web?"

RÉSUMÉ. *Avec la démocratisation du web, la quantité d'informations transmises par des utilisateurs vers des sites web a explosé. Actuellement ces données sont stockées sur les serveurs des entreprises et une fois transmises, deviennent inaccessibles pour leurs auteurs. Nous nous proposons de démontrer qu'en traçant les actions de l'utilisateur selon des modèles de traces bien définispermettant ainsi aux utilisateurs de conserver leurs informations, nous pou- vons faire bénéficier de ces informations à la fois aux utilisateurs mais également aux entreprises destinataires de ces informations. Dans ce papier, nous présentons un modèle de traces incluant la qualification sémantique de pages web et une solution à base d'extension de navigateur qui propose des outils de traçage. Un de nos objectifs est de proposer des méthodes d'analyse permettant de répondre à des questions du type : "Sur quelle page web ai-je obtenu des informations à propos d'Inforsid ?".*

KEYWORDS: *trace for the user, vrm, web trace model*

MOTS-CLÉS : *traçer pour l'utilisateur, vrm, modèle de trace web*

1. Introduction

Advancements in computer technology have largely reduced the hardware costs especially for storage. Currently, almost all web sites have the capacity to record much information for and about their users. Furthermore, on one hand, many tools (CMS, CRM, data mining) have been developed to allow sites to obtain, store, access, manage and exploit data provided by their users. On the other hand, a user has no tools to manage her data submitted online. As a result, the access to information is increasingly unbalanced since, paradoxically, the more computers capacities increase, the more the user loses control over data she submits on the web. To illustrate this point, let us think about: who can make an exhaustive list of information explicitly transmitted using web forms, when they have been transmitted, to who, and for which purpose?
Without contesting the right for sites to store data, it seems natural and highly complementary, to enable the user to record and structure her own data. In addition, the user may take advantage of migrating some data to her personal applications. For example, in the context of an online purchase, the transaction amount can be transmitted to her personal account manager and an event can be set in her calendar to remind the delivery date. Yet, this is not currently possible.
Such arguments seem to reach the point of view developed by the Berkman Center For Internet & Society at Harvard University when they present the notion of Vendor Relationship Management (*VRM*) (Havard, 2007) which is the customer-side counterpart of the more known Customer Relationship Management (Wikipedia, 2013). While the expected benefit of a CRM are turned to the vendor, the VRM focuses on individuals. Five main properties have been identified (Havard, 2007). Our objective is to go further, proposing a tool compliant with those which are related to our context: “Customers must be the points of integration for their own data”; “Customers must have control of data they generate and gather. This means they must be able to share data selectively and voluntarily”; and “Customers must be free to express their demands and intentions outside of any company’s control".

To move towards this goal, in this paper, we present our first proposal, ($Adnosco$) a tool which enables to model, trace, store, search and manage submitted data. Due to lack of space, we do not intend to present all the potential uses for stored user traces. We will focus rather on the benefits of storing, at the client side, data submitted to web sites. Local storage of data submitted through web form fields raises several problems starting with their acquisition. Some operational solutions, developed in other contexts, already exist (e.g., Lazarus (Interclue, 2011), Dashlane (Dashlane, 2013)), and we look at them as proofs of concept. We mainly focus on using the user traces for the input help. Indeed, **completion** and **pre filling** are functionalities which have already shown their ability to induce significant productivity gains in professional contexts and which can rely on user data.

In many cases, it is interesting for the user to reuse the already submitted data. That is the spirit in which many browsers propose values. But they limit their proposals to values previously typed in the same field. Our first syntactical approach proposes the

completion and pre filling functionalities, but it goes a step ahead by exploiting stored personal data rather than just data already filled in by all users. However, when filling in a web form for the first time, data can only come from other web forms and in such case, the syntactic approach has a weak accuracy due to the high heterogeneity of web forms. To improve precision, we propose to manage heterogeneity by introducing a semantic based-approach. Lastly, considering that a user must often fill in different web forms to achieve a higher goal, we introduce the concept of "activity" that makes concrete an aspect of the "user's context". The intuition is simple: the data involved in an activity are often repetitive. For example, a trip planning activity goes through different web sites for travel, accomodation, etc. Again, it is important to keep in mind that these assistance tools are devoted to the user who is $infine$ the one that can judge the coherence of the data to be submitted. Hence any proposed tool should not be intrusive. It is up to the applications and services in concern to enforce their rules in order to obtain consistent data.

This paper is outlined as follows: section 2 gives a formal definition of web forms and presents a motivating scenario. Section 3 shows a generic definition of completion and pre filling functionalities. Sections 4, 5 and 6 show how syntactic, semantic and activity based approaches deal with user's data. Section 7 presents a theoretical evaluation of our proposal. Finally related works are discussed in Section 8 before concluding.

2. Background

Since the web forms are central to our study in this paper, it is important to define them formally. We abstract from presentation and language (HTML version, embedded code...) as well as from all technical points to focus only on information related to our issue.

Definition 1 (Web form)
A web form *wf is a tuple* $\langle uri, fields, tt \rangle$ *where*

– uri *is an URI*

– $fields(wf)$ *is a finite non empty set of* n *fields* $\{f_{wf,1}, f_{wf,2}, ..., f_{wf,n}\}$
A field is a triple $\langle name, type, value \rangle$.

– tt *is the Transaction Time (i.e. the submission time stamp).*
tt *is set to* $null$ *until the form is submitted.*

3. Proposal General Sketch: Functionalities

Our general concern is about the loss of control of a user over the information she submits on the web. This paper does not claim to address this problem in its generality. Rather we focus on a first step which is to acquire these data, to store them, to structure them and to exploit them in such a way that the user obtains clear

advantages. A beneficial side effect could be to raise individual user awareness about transmitted data potential.

Assuming these data acquired, we propose to help the user to fill in web forms via three functionalities to enhance the user experience and to enable significant productivity gain: restoring values of a web form to get it exactly as it was submitted (given a submission date); proposing possible completions to the user while she is trying to fill in a form field; and, pre-filling fields when the user has not yet entered any value for the form field and for which the server has not proposed default values. These functionalities are well known in many other applications, but, even considering recent advances of our browsers, when not completely absent, they are available in very marginal situations. Having access to the whole user data, $Adnosco$ can offer an extended and powerful pre-filling and completion functionalities. Let us briefly present how we consider these functionalities in $Adnosco$. Then we will detail the different methods for the completion functionality in the three next sections.

Restoring values. Restoring webform field values as they were submitted at a certain date is not a so complicated task as far as filled web forms are stored. The Dashlane (Dashlane, 2013) and in some measure the Autofill Form Firefox plugins (Interclue, 2011) propose such a functionality restricting to recent submissions which is already really helpful.

Completion and pre filling are more complex functionalities. Following this intuition, we propose a set of methods each of them producing a set of relevant values ordered according to a partial pre-order encoding their relative relevance. Each method answers the same question in different ways. The general question is "*in the context of the considered web form (already filled values, already typed value in the current field...) which values to propose to the user?*". A complementary question is, "*how to order these results?*". The proposed answers are stored in a simple triple $\langle$methodName, $ValueSet$, $\leq \rangle$. We will then obtain as many of these triples as there are proposed methods. Methods can be ordered according to their assumed accuracy. Thereby, the global results of all invoked methods are ranked in the list L: from the one assumed to be the most accurate, to the last. This order can be determined a priori by the application or set up by the user.

The next section is devoted to different methods to obtain relevant values and associated pre-orders. We mainly explore three possible ways.

1 - *Syntax-based approach* helps the user filling in the web form fields based on syntactic criteria. In this case, proposed values have to be selected with respect to syntactic consideration only, i.e., same field or same type. 2 - *Semantics-based approach* brings semantic notions within the picture to obtain more accurate results. 3 - *Activity-based tool* proposes to take into account some user's context introducing the notion of "activity". The intuition is to know what the user is doing to better understand how its current web form filling is related to previous ones. at the end of section 6 , we will illustrate thanks to our scenario, the calculation of the list L and how it is exploited to display the list of possibilities when trying to fill in a field.

4. Extracting Values and Relevance Order based on Syntax

We propose two different syntactic methods to extract values relevant to a field of a web form. The first is simply to consider only what the user has already submitted through the same field during a previous submission of the same web form while the second increases the scope by searching all available values in fields having the same type, regardless the web form where they appear.

4.1. *Location based approach: same web form, same field*

4.1.1. *The value set*

can be obtained looking only at values already filled in the same field. More formally, the value set can be defined as: $ValueSet^{SF}_{syntactic}(wf, f, c, W)$ where wf is the web form under concern, f is the field under focus, c is the actual field value ($null$ if none is present), and W is the set of web forms to take into account. Except in some particular cases, usually, this last is set to WF (the set of all known web forms).

4.1.2. *Associated preorder candidates*

Many preorders are natural candidates to qualify the relevance of obtained values:

– $\leq_{sim}$ is a preorder over values which is obtained considering similarity between equivalent web forms.

– $\leq_{freq}$ is a preorder over values which is obtained considering the frequency of a term.

– $\leq_{trans}$ is a more simple pre-order which just considers the transaction times.

– $\leq_{nat}$ is the simplest among proposed pre-orders. It does not pay any attention to the web form but it focuses on the natural order of values according to their type (alphabetical, numerical, etc.).

To close this consideration on orders, one could be interested to combine them. It is possible, and for example, $\leq_{trans-sim}$ denotes the composition where two values are first ordered using $\leq_{sim}$, and, in case of equality are ordered using $\leq_{trans}$. According to the resulting order, the top corresponds to the most recent value among those which appears in most similar web forms.

4.2. *Type based approach*

4.2.1. *The value set*

is obtained considering only the values of the same type than the type of the concerned field, in previous submitted web forms. The number of presented values is higher than for the syntactic same field method

4.2.2. *Associated pre-order candidates*

Excepted for the similarity approach which requires equivalent web forms, previously introduced notions can be used here. Conversely to $\leq_{trans}$ and $\leq_{nat}$) which can be used without any modification, due to the fact that a value may appear more than one time in a web form, $\leq_{freq}$ which is based on frequency, has to be adapted.

5. Extracting Values and Relevance Order based on Semantics

As we saw in the previous section, syntactic methods reach their limits very quickly. For the first one, its research field is too limited to enable to find the desired value. On the contrary, for the second one, it enlarges considerably the result set as it brings a very (too) large number of values, no order being able to bring out relevant ones. Clearly, the type of data is not a sufficiently accurate criteria to distinguish between values. Better structuring is needed to improve performance. Unfortunately, unlike the site that produces the form, a user does not have the information she needs about the structure of the data contained in a form. To break the deadlock, we propose to bring semantics (ontologies) into the picture. Obtaining ontologies and semantic qualification will also be a problem. There are many possibilities. They can be built by the user, but this is a huge work for a single person. They can be provided by the sites as they provide CSS style sheets, but we are far from that. Or they can be built and shared by users communities and associations. Whatever, before thinking about how to obtain them, we have to define them and to evaluate how interesting they are, for our objective.

5.1. *Semantic qualification of web forms*

To semantically qualify a web form, one can think to link web form fields to concept properties. However, in presence of multiple instances of the same concept within the same web form, this technique does not allow to distinguish them. For instance, in figure 1 there are information about two addresses with two names, counties, ... A simple syntax based assistance cannot distinguish between the two field sets concerning the two addresses. To obtain a semantic qualification with a higher structuring power, we propose to introduce the notion of *materialized concepts*. A *materialized concept* links a group of fields to a concept of an ontology, associating each field to a property of this concept.

5.1.1. *Formal definition*

Definition 2 (Materialized Concept) *A* materialized concept associated to a web form wf *(definition 1) is a quadruple* $\langle name, concept, wf, Corr \rangle$ *where*

– $name$ *is the name of the materialized concept which is a simple string. It should be unique for a web form.*

– $concept$ is a concept of an ontology.
The set of properties associated to that concept are noted $Props(concept)$.

– wf is the web form to which the materialized concept is associated.

– $Corr$ is a set of triples $\langle fieldName, op, property\rangle$ precizing how fields of the web form materialize the associated $concept$:

- $fieldName$ is the name of a field of wf,

- op is an operation which enables to deduce the field value considering the value of a property (in this paper, for the sake of simplicity, we consider only the identity (=) symmetric operation).

- $property$ is a property of the concept $concept$.

Unsurprisingly, a semantic qualification of a web form may involve more than one materialized concept.

Definition 3 (Semantic qualification of a web form) *A* semantic qualification sq of a web form wf*(definition 1) is a triple $\langle name, wf, MC\rangle$, where*

– $name$ is a unique name of the semantic qualification.

– $wf \in WF$ is a web form concerned by the semantical qualification.

– MC is a set of materialized concepts associated to the web form wf, i.e. $\forall mc \in MC$, $mc.wf = wf$.

The semantic qualification of a web form is done independently of the embedded values. In other terms, if a semantic qualification is done for a web form wf then it applies to any other equivalent web form.

Notations

– $SQ(wf)$ denotes the set of all known semantic qualifications of the web form wf, i.e. $SQ(wf) = \{sq : sq.wf = wf\}$.

– SQ denotes the set of all known semantic qualifications, i.e. $SQ = \bigcup_{wf \in WF} QS(wf)$.

5.2. *Using semantical qualification to extract value set*

Let wf be the web form under concern, $f \in wf.fields$ the field having the focus, and c the current value of the field.
$ValueSet_{Sem}(wf, f, sq, c, W)$ is the set of elements of the format $qs'.wf'[f']$ [1], such that:

1. reminder: in this paper, for the sake of simplicity, we only consider equality operation.

– f is semantically associated to a property of a materialized concept mc.
$\exists mc \in sq.MC$, $\exists al \in mc.Corr$: $al.fieldName = f.name$, and

– according to some semantic qualification sq', there exists fields semantically associated to the same property of the same concept.
$\exists sq' \in SQ$, $\exists mc' \in sq'.MC$, $mc'.concept = mc.concept$ and $\exists al' \in mc'.Corr$, $al'.property = al.property$, and

– among fields so qualified, we are interested into those which start with the value c
$\exists wf' \in EQ(sq'.wf) \cap W$: $((wf'[al'.fieldName]$ starts with $c)$ or $(c = null))$.

5.3. *Associated preorder candidates*

The resulting set can be ordered considering many criteria. $\leq_{freq}$, $\leq_{trans}$ and $\leq_{nat}$ are here again good candidates.

5.4. *Illustration scenario*

To illustrate Adnosco syntactic and semantic assistants efficiency, let us present the following scenario. John Smith is living in Lyon. He decides to offer a gift to his daughter Alice who is married and lives in Lille. John is used to buy his gifts on Internet. He connects to the *pc21.fr* website and fills in the main web form with his personal information, his address in Lyon considered as the billing address, the personal information and address information (considered as the delivery address) for his daughter. John has just downloaded Adnosco. He decides to experiment it and defines four materialized concepts framed respectively in red (Customer), orange (InvoiceAddress), blue (DeliveryAddress) and cyan (Recipient), one for each subset of fields described below. He also links semantically his personal information (red frame) with his address information(orange frame). He does the same link between her daughter personal information and address (see figure 1).

Later, John travels to Lille. For his return travel, he plans to take a train that arrives at 23:00 to Lyon. He then decides to book a taxi to get home from the railway station. To do this, he goes to the *taxis-lyonnais.fr* web site and fills in it. John never visited this site but the site's webmaster has linked semantically the fields *name (Nom ou Société)* and *e-mail* to the ontological concept *person* and the field *telephone* to the *address* ontological concept as it is bound to the physical address of somebody. The two sections *Departure address (Adresse de départ)* and *Arrival address* (figure 3) have been also qualified semantically and linked to the concept *address* using two distinct materialized concepts. John types 's' in the *name* field. As shown on figure 2, Adnosco displays in the context menu syntactic then semantic choices. The names *Smith* and *Snoopy* are proposed in the case of syntactic completion as they are both stored in the Adnosco data storage as possible values for the *name* field of the *pc21*

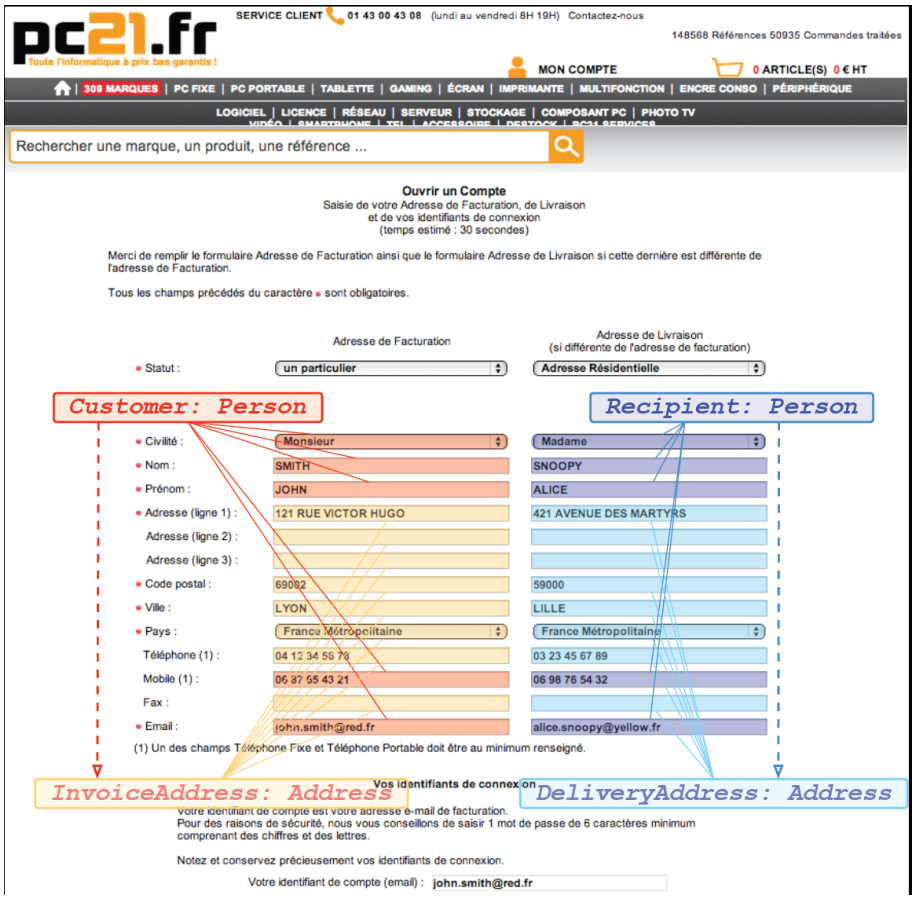


Figure 1: Materialized concepts on a webpage (pc21.fr)

web form. They are also proposed by the semantic assistant as they are the two values associated to the attribute *name* of the concept *person*. John chooses his name in the semantic set of choices. Immediately, the field *e-mail* is filled automatically by John's e-mail as in the storage the name and the e-mail are associated to the same materialized concept. Additionnally selecting *Smith* as the user name, implies the automatic fill in of the *telephone* field. In fact, as the *telephone* field is linked semantically to the same materialized concept as for the *address* fields in the *pc21.fr* web form, Adnosco proposes the phone number stored previously for John Smith when he filled the *pc21.fr* web form. Finally when John tries to fill in the *city (*ville)* field in the *Arrival address* section (see figure 3, the semantic assistance proposes 'Lyon' and 'Lille' as both are city names. The syntactic assistant doesn't give any proposal as John is visiting the *taxi-lyonnais.fr* web form for the first time.

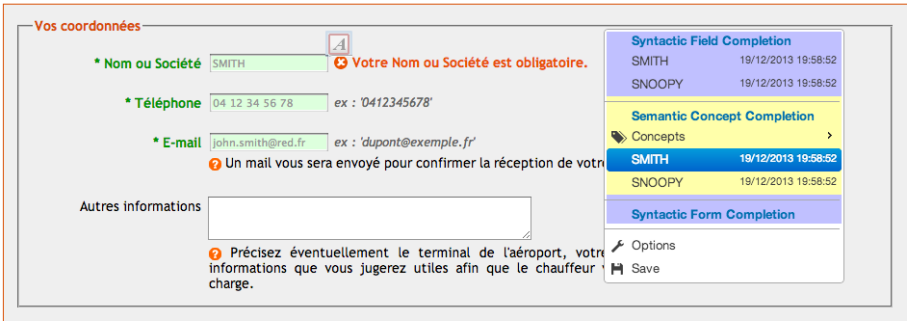


Figure 2: Illustration of Semantic assistance based on semantic qualification dependencies (taxi-lyonnais.fr). Adnosco is called for the field with the "A" icon. The green fields are the automatic semantics based completion proposals corresponding to the selected value in the menu.

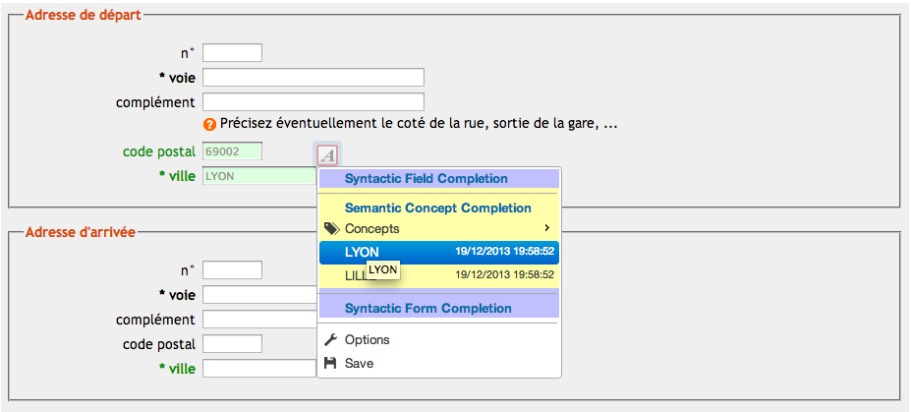


Figure 3: Illustration of Semantic assistance based on semantic qualification (taxi-lyonnais.fr)

6. Extracting Values and Relevance Order considering Activity

The previous section shows that semantics is helpful to propose relevant values to the user. We introduce the activity notion to make possible to cross data from one web form to another as far as they are part of the same activity, and this without any concern about the provider to which they have been uploaded nor about time from last connected data upload.

Intuitively, an activity polls many forms for a specific purpose. The concerned forms may have been designed specially for this purpose (ex.: set of web forms of

a flight booking site), or more interestingly have been developed independently (ex. flight booking site + car rental site + hotel site) and used together in a dynamic manner. To formalize the information overlap between forms, they are aggregated within an identified activity which corresponds to a particular context of use. For this purpose, we rely on materialized concepts which have been introduced to semantically qualify web forms.

Definition 4 *Activity*
An activity A *is a tuple* $\langle name, W_A, Q, C\rangle$ *where*

– $name$ is the name of the activity. We assume it to be unique (i.e. there is no different activities sharing the same name).

– W_A is the set of web form gathered in the activity $name$.

– Q is a set of semantic qualifications such that there is at most one semantic qualification per web form
$\forall(sq_1, sq_2) \in SQ^2$, if $sq_1.wf = sq_2.wf$ then $sq_1 = sq_2$.

– C is a set of correspondences between materialized concepts present in the semantic qualifications that belong to Q.
A correspondence between materialized concepts is defined as a tuple $\langle sq_1, mc_1, sq_2, mc_2\rangle$ such that

- *$\forall i \in \{1, 2\}$, $sq_i \in Q$*
- *$\forall i \in \{1, 2\}$, the materialized concept $mc_i \in sq_i.mc$*
- *$mc_1.concept = mc_2.concept$.*

This condition could be relaxed considering mapping between ontologies, but here it is out of scope of this paper. For the sake of simplicity, we restrict ourselves to consider only concepts of the same ontology.

Correspondance obtained by transitivity are automatically added.

$(wf, f) \simeq_{ad} (wf', f')$ denotes the fact that in the context of an activity defined by ad, the fields f and f' belonging respectively to the web forms wf and wf' are related to the equivalent materialized concept according to the activity ad. The interpretation of this correspondence is that, normally, the materialized concepts mc_1 and mc_2 should lead to instances with the same property values, i.e. identical values for their corresponding fields in the web form instances. The user still has the ability to waive this rule. For this he simply enter the values she wants, by ignoring those proposed by our system. We don't want to impose anything to the user, letting him fully responsible about data she communicates. The problem of checking integrity constraints is left to the sites.

Definition 5 (Activity instance)
An activity instance *is a a tuple* $\langle name, ad, WF_{ai}, st, ct\rangle$ *where*

– $name$ is the name of this instance, assumed to be unique,

– *ad is the activity on which this activity is based,*
– *WF_{ai} is a set of web forms involved within this instance activity.*
– *st is the starting time of the activity instance, and*
– *ct is the closing time of the activity instance.*

It is interesting to note that in an activity instance all web forms present in its associated definition do not have to be present and reversely, some web forms may be embedded into the activity by the user even if they are not semantically qualified.

6.1. *Extracting value sets with respect to an activity*

Here the objective is to extract values strongly related to the current activity. By definition, they are semantically related and so already proposed by the peviously presented semantic approach, but they are much less numerous. To be more precise, their number does not depend on all uploaded web forms but only on those belonging to the activity instance. We also take advantage of expressed correspondences over materialized concepts. The extracted value according to these rules set is noted $ValueSet_{Act}(wf, f, c, ai)$ where wf is the web form under concern, f is the field under focus, c is the actual field value ($null$ if none is present) and ai is the current activity instance. This method selects the values of fields linked to f through materialized concepts and activity correspondences, and whose values are compatible with c. More formally,
$ValueSet_{Act}(wf, f, c, ai)$=$\{wf'[f'] : wf' \in ai.WF_{ai}$ and $wf.f \simeq_{ad} wf'$ and $wf'[f']$ starts with $c\}$.

7. Evaluation

In this section, we evaluate the time gain in filling web forms when using $Adnosco$. Let us consider the following situation. A user has to fill in 3 web forms in order to book a journey. Let's say the first web form is about 2 traveler persons and contains 6 fields (Name, Surname,City)*2, the second webform is about travel information, containing 8 fields ((Name, Surname)*2, Outbound Dearture and Arrival date and Inbound Dearture and Arrival date) and the 3^{rd} one is about the car rental containing 5 fields (Name, Surname, Departure Date, Arrival Date, Car type). We also consider that the Name and Surname fields from the first 2 webforms are in correspondence through an Activity instance as well as the first Name and Surname field from the second webform with the Name and Surname of the third one. Without any assistance she has to fill the $6+8+5 = 19$ fields in the three web forms. Considering 0,4 seconds needed per character [2], 5 characters in average per word and 2 words in average per field, that would take at least 76 seconds. With a syntactic completion assistance after

2. Words per Minute, `http://en.wikipedia.org/wiki/Words_per_minute`

typing in the first characters, the system provides the correct value. This reduces to 4 the average character number to type per field and thus give a theoretical period of 30,4s. The semantics based assistance increases the precision of the recommended values and thus decreases the number of characters to type in a field to 2 giving 15,2 seconds of filling time. With an activity based pre filling, the time needed to fill the first web form remains the same. For the second web form, only the dates will be to precise (4 fields instead of 8), as for the third web form, the only field to fill in will be the $Cartype$. The total theoretical time necessary to fill in the three web forms will be reduced to $6 \times 2 \times 0,4 + 4 \times 2 \times 0,4 + 2 \times 0,4 = 8,8s$. This represents a theoretic gain of 88%.

To generalize, we estimate this gain through calculations. First we announce a set of hypothesis:

– the average length in characters of a word is LC (5 in the previous example)

– the average word count in a web form field is WC (2 in the previous example)

– the average number of character to type before getting a correct completion is TLC (4 in the previous example)

– the percentage of correct pre-filling over the set of web forms in a given activity is CPF (70% in the previous example)

We can say that if we have N web forms to fill in in a given activity, with n_i fields each ($i = 1..N$) and t is the average time to fill in a field, we need $\sum_{i=1}^{N} (LC * WC * n_i * t)$ time to complete the N web forms. Adding the assistance, this time is reduced to $\sum_{i=1}^{N} (TLC * (1 - CPF) * n_i * t)$. Without semantic assistance $TLC = LC * WC$ and without activity based assistance, CPF is 0.
The estimation of parameter values obtained on our simple example have to be verified nevertheless through experimentations on real data sets considering different activities.

8. Related Work

This section covers two domains : **applications for managing personal information** and **the alignment of data sources**.

There are several fields that tackle the management of user data. One of them is gathered around the online identity community and places the end-user at the center. They relay all communication between identity providers and service providers through the user's client (Bramhall *et al.*, 2007) (Cameron, 2005) (Marc Goodner, 2008) enabling people to have and employ a collection of digital identities. The Information Card metaphor is implemented by Identity Selectors like Windows CardSpace (Cameron, 2005) (Nanda, 2007) and the Higgins project (Higgins, 2007). An Identity Selector system generally provides the user with an interface to create and manage personal information cards. These works mainly provide solutions to avoid unsuper-

vised spreading of user data, but don't help her track and reuse information filled in web forms.

Another category of works includes applications that follow user actions inside web browsers, such as the Firefox plug-ins: $CoScripter, Lazarus, AutofillForms$ and $PrivacyDashboard$. While $CoScripter$ translates user actions in plain text, $Lazarus$ stores web form data and enables to refill the form as it was submitted at a given date. $AutofillForms$ is announced to be the closest one to $Adnosco$ but actually it doesn't work and has very little documentation. $PrivacyDashboard$ provides a control interface to check what kind of information is sent to which website. The $Collusion$ and $Moluti$ plug-ins enable to analyze surf surveillance and history, while (Worlfram|Alpha, 2014) provides Facebook data analysis.

Concerning the webform semantic qualification, the alignment task, also known as matching, has been studied for many decades (Batini *et al.*, 1986). Traditionnally, the structured data sources involved in alignment can be ontologies (Euzenat et Shvaiko, 2007), schemas (Bellahsene *et al.*, 2011) , or entities (Talburt, 2011). Alignment tools usually combine different similarity measures (e.g., instance-based, terminological, lexical, constraint-based) applied to the elements of the data sources (e.g., concepts, properties, instances). In ontology alignment, researchers compete during the annual OAEI challenge using various datasets to demonstrate the effectiveness and performance of their tools (Euzenat *et al.*, 2011).

A few works have focused on matching unstructured data sources, such as web forms. The SMB approach deals with the matching of two web forms from the same domain (converted in the OWL format) (Marie et Gal, 2008). In a similar fashion, the UIUC repository collects query interfaces to help understanding the modelling and integration of web databases (UIU, 2003) . Zhang et al have used the UIUC collection to discover a hidden syntax from web forms (Zhang *et al.*, 2004). At this point we are tending more towards ontologies proposed on $schema.org$ to align the webforms with.

Although this paper mainly presents the foundations of $Adnosco$, our tool is also designed for discovering semantic links between a web form and an ontology using similarity metrics. Contrary to all these alignment approaches, $Adnosco$ does not perform any alignment between two structured data sources (e.g., ontologies, schemas) or two unstructured data sources (e.g., web forms). The additional semantic layer proposed in our approach includes materialized concepts, which bridge the gap between a structured data source (an ontology) and an unstructured one (a web form). Besides, a materialized concept takes into account the instances of a web form, so that related data are stored together and can be proposed later with effectiveness. The syntactic and semantic assistants extend comparable solutions (Dashlane, 2013 ; MIT, 2013) to values issued from other websites and to more precise and rich propositions based on webform semantic qualification and the use of materialized concepts to handle multiple concept instances on the same web form.

9. Conclusion

In this paper, we have introduced $Adnosco$, a **user-centric personal data tracer and manager** that allows the user to store her own data submitted online through web forms. $Adnosco$ is also able to store the trace model including the organization to which the information has been transmitted, transaction timestamp and the mean of transmission (web form or any other data transmission). Besides, $Adnosco$ can be considered as a non-repudiation mean in case of lost web form submission. This paper mainly focuses on one of the advantages of $Adnosco$: its ability to assist efficiently the user in **filling in forms** on her navigator. To this end, when the user attempts to fill in a value, data is extracted from $Adnosco$ repository and proposed to her, in an order that corresponds to her chosen configuration. Three methods to extract data and their possible data orders has been proposed: syntactic, semantic and activity based. The **syntax-based** method proposed values corresponds to all data beginning with the same characters than the current filled in field, while the **semantic-based** method extracted data corresponds to data that is semantically qualified similarly than the current field; finally, the **activity-based** extraction is more selective as it limits the proposals to previous data filled in in the forms that belong to the same activity or more selectively to the same instance of some activity. Besides, activity-based assistance is empowered thanks to the novel notion of materialized concepts and their established correspondences, that *in fine*, precises the set of proposed values.

In the future, we plan to extend field correspondences to other operation types to increase the expressiveness of $Adnosco$ and facilitate the **discovery of semantic links** between data. Furthermore, we are working on an automatic tool to help users discover materialized concepts between an ontology and a web form. We are exploring two methods to fulfill this goal. The former matches the elements of the new web form directly with the existing materialized concepts. The latter first aims at detecting the web forms already matched in $Adnosco$ which are semantically close to the new web form, and then to perform a fine-grained matching between the new web form and the ontologie(s) which are matched to the closest web forms. To evaluate both methods, we will propose a **benchmark** whose datasets are composed of web forms from related domains (e.g., flight booking, hotel booking, car rental), ontologies, and the materialized concepts between them. Such a benchmark will allow us to confirm experimentally the results presented in this paper.

10. References

Batini C., Lenzerini M., Navathe S. B., "A Comparitive Analysis of Methodologies for Database Schema Integration.", *ACM Computing Surveys*, vol. 18, num. 4, 1986, p. 323-364.

Bellahsene Z., Bonifati A., Rahm E., *Schema Matching and Mapping*, Springer-Verlag, Heidelberg, 2011.

Bramhall P., Hansen M., Rannenberg K., Roessler T., "User-Centric Identity Management: New Trends in Standardization and Regulation", *Security Privacy, IEEE*, vol. 5, num. 4,

2007, p. 84-87.

Cameron K., "The Laws of Identity", http://msdn.microsoft.com/en-us/library/ms996456.aspx, 2005.

Dashlane, "Dashlane", https://www.dashlane.com/download/Dashlane_IFOP_release_2013-03-26_en.pdf, 2013.

Euzenat J., Ferrara A., van Hage W. R., Hollink L., Meilicke C., Nikolov A., Ritze D., Scharffe F., Shvaiko P., Stuckenschmidt H., Sváb-Zamazal O., dos Santos C. T., "Results of the ontology alignment evaluation initiative 2011", Shvaiko P., Euzenat J., Heath T., Quix C., Mao M., Cruz I. F., Eds., *OM*, vol. 814 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2011.

Euzenat J., Shvaiko P., *Ontology matching*, Springer-Verlag, Heidelberg (DE), 2007.

Havard, "Vendor Management System", http://cyber.law.harvard.edu/projectvrm/, 2007.

Higgins, "Higgins, open source identity framework", http://eclipse.org/higgins/, 2007.

Interclue, "Lazarus, Mozilla plugin", http://getlazarus.com/, 2011.

Marc Goodner A. N., "Identity Metasystem Interoperability Version 1.0", http://www.oasis-open.org/committees/download.php/29979/identity-1.0-spec-cd-01.pdf, 2008.

Marie A., Gal A., "Boosting Schema Matchers", *OTM Conferences (1)*, Berlin, Heidelberg, 2008, Springer-Verlag, p. 283–300.

MIT, "openpds", http://openpds.media.mit.edu/, 2013.

Nanda A., "Identity Selector Interoperability Profile V1.0", http://download.microsoft.com/download/1/1/a/11ac6505-e4c0-4e05-987c-6f1d31855cd2/Identity-Selector-Interop-Profile-v1.pdf, 2007.

Talburt J. R., *Entity Resolution and Information Quality*, Elsevier, 2011.

"The UIUC Web Integration Repository", Computer Science Department, University of Illinois at Urbana-Champaign. http://metaquerier.cs.uiuc.edu/repository, 2003.

Wikipedia, "Customer Management System", http://en.wikipedia.org/wiki/Customer_relationship_management, 2013.

Worlfram|Alpha, "Personal Analytics for Facebook", http://www.wolframalpha.com/facebook/, 2014.

Zhang Z., He B., Chang K. C.-C., "Understanding Web query interfaces: best-effort parsing with hidden syntax", *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, SIGMOD '04, New York, NY, USA, 2004, ACM, p. 107–118.

Proposition d'une démarche de type IDM pour la construction d'outils d'exécution de processus

Sana Mallouli[1] — Saïd Assar [2,1]— Carine Souveyet[1]

[1] Université Paris 1 Panthéon Sorbonne, Centre de Recherche en Informatique, 90, Rue de Tolbiac F-75013 Paris
sana.mallouli@malix.univ-paris1.fr
carine.souveyet@univ-paris1.fr
[2] Institut Mines-Télécom, École de Management, Dépt. Systèmes d'Information, 9, Rue Charles Fourrier F-91011 Evry
said.assar@telecom-em.eu

RÉSUMÉ. *L'ingénierie des systèmes d'information fait appel à de multiples langages pour modéliser, programmer et manipuler divers artéfacts tout le long du cycle de développement. Ces langages sont généralement supportés par des outils logiciels. La construction de ces outils est une tâche complexe menée souvent de manière ad-hoc. L'exploitation adéquate des techniques de méta-modélisation a le potentiel faciliter cette tâche. Cependant, pour un langage de modélisation de processus, la prise en compte de sa sémantique d'exécution dans un méta-modèle n'est pas encore au point. Dans cet article, nous présentons une démarche basée sur l'usage de méta-modèles événementiels et de règles de transformation pour décrire la sémantique d'exécution d'un modèle de processus et en dériver une architecture d'outil d'exécution. Cette démarche est expérimentée avec un langage de modélisation orienté intention et le résultat est comparé avec des travaux antérieurs.*

ABSTRACT. *Information System engineering involves multiple languages for modeling, programming and handling various artifacts throughout the development cycle. These languages are usually supported by software tools. The construction of these tools is a complex task often done in an ad-hoc manner. Proper use of meta-modeling techniques has the potential to facilitate this task. However, for process modeling language, properly expressing and exploiting process execution semantics is still a challenging issue. In this paper, we present an approach in which process execution semantics are expressed using an event meta-model, and a set of transformation rules are defined and applied in order to derive the technical architecture of an execution tool. This approach is tested on the case of an intention-oriented modeling language, and the result is compared with previous work.*

MOTS-CLÉS : Ingénierie des langages logiciels, méta-modélisation, exécutabilité d'un modèle, sémantique d'un modèle de processus, exécution de processus.

KEYWORDS: Software languages engineering, meta-modeling, model executability, process model semantics, process execution.

1. Introduction

Dans l'essor du génie logiciel et de l'ingénierie des systèmes d'information, les langages jouent un rôle prépondérant. Que ce soit pour programmer, modéliser, transformer ou interroger, ces langages sont essentiels pour construire, manipuler et raisonner sur des artéfacts tout au long du cycle de vie des systèmes. Ceci nécessite la construction d'outils logiciels divers et variés pour supporter ces langages et rendre concrètement possible leur utilisation. Ce besoin s'est accentué avec le développement de l'ingénierie dirigée par les modèles (IDM) qui, justement, prône l'usage intensif de modèles tout le long du cycle de conception et de développement des systèmes (Favre *et al.*, 2006). L'édition et la vérification d'un modèle, la transformation d'un modèle vers un autre et la génération du code final deviennent ainsi des tâches essentielles qui doivent être spécifiées, formalisées et mises en œuvre dans des outils logiciels qu'il faut construire (Jouault *et al*., 2009).

L'objectif de l'ingénierie des langages est de spécifier les langages informatiques et de construire des outils logiciels de support (Favre *et al.*, 2009). Pour spécifier un langage de modélisation, on fait appel à la méta-modélisation (Atkinson et Kühne, 2003)(Sprinkle *et al.*, 2011). Un méta-modèle est un modèle d'un langage de modélisation. C'est une représentation, généralement graphique (mais pas forcément), des concepts sous-jacents au langage, des liens entre ces concepts ainsi que d'éventuelles contraintes que doivent satisfaire ses instances. Dans la terminologie des langages de programmation, un méta-modèle est l'équivalent de la syntaxe abstraite d'un langage (Kleppe, 2009a). La sémantique d'un langage concerne le sens que peuvent prendre les constructions de celui-ci lorsqu'elles sont instanciées dans un modèle (Harel et Rumpe, 2004). Elle n'est que partiellement prise en compte par un méta-modèle (Sprinkle *et al.*, 2011). Dans l'univers des langages de programmation, le problème de l'expression de la sémantique s'y pose dans des termes similaires. En effet, les méta-compilateurs exploitent des langages basés sur la notation Backus-Naur Form (BNF) pour spécifier la syntaxe et s'en servir pour dériver des outils (Kleppe, 2009b). L'aspect sémantique est relégué à un traitement par programmation directe, même si la compilation a introduit des approches innovantes pour organiser cette tâche, telle que la grammaire des attributs (Paakki, 1995). Et depuis peu, une communauté de chercheurs appelle explicitement à rapprocher l'univers de l'IDM et celui de la compilation (Jézéquel *et al.*2012).

L'expression explicite, éventuellement formelle, de la sémantique d'un langage est une problématique importante dans l'ingénierie des langages. Elle répondrait à plusieurs besoins tels qu'une spécification plus rigoureuse d'un langage, l'analyse et la validation de cette spécification, ou encore la génération automatique d'outils support pour le langage (Bryant *et al.*, 2011). Lorsque cette sémantique se limite à des contraintes de validité que doivent respecter les instances d'un modèle, elle est appelée sémantique statique. Elle peut s'exprimer avec un langage de règles au niveau du méta-modèle, tel le langage OCL (Object Constraint Language). Mais

pour un langage de modélisation de processus, avec des concepts tels que flot de données, état et transition, la sémantique désigne alors un comportement dynamique qui correspond à la manière avec laquelle un processus s'exécute. Cette sémantique d'exécution est difficile à capturer dans un méta-modèle, elle est néanmoins indispensable pour construire des outils pour exécuter un modèle de processus.

Dans cet article, nous développons une démarche d'ingénierie des langages pour construire un outil d'exécution de modèles de processus. A l'aide d'une notation événementielle, nous complétons la spécification statique avec un schéma dynamique qui capture la sémantique d'exécution du modèle. Grâce à des règles de transformation, ce schéma est converti en une architecture logicielle d'un outil d'exécution dans un environnement cible. Cette architecture exploite des motifs génériques d'exécution (des « patterns ») de type publier/souscrire (Eugster *et al.*, 2003). Ces motifs architecturaux sont adaptés pour la programmation de systèmes asynchrones faiblement couplés (Hinze *et al.*, 2009), et permettent de transcrire correctement la sémantique d'exécution événementielle d'un modèle de processus.

Le reste de cet article est structuré comme suit. La section 2 est un bref état de l'art concernant l'expression de la sémantique, et les méta-outils d'ingénierie des langages. La section 3 présente notre méta-démarche et introduit la modélisation événementielle de sémantique d'exécution. Les règles de transformation sont ensuite présentées dans la section 4. Dans les sections 5 et 6, cette démarche est appliquée à un langage de modélisation intentionnelle des processus et en guise d'évaluation, le résultat obtenu est comparé avec des travaux antérieurs ayant le même objectif. L'article se termine par une conclusion qui évoque les limites et les travaux futurs.

2. État de l'art

La question de la sémantique d'un langage se retrouve dans deux courants de recherche complémentaires. Le premier est directement lié aux méthodes et techniques de spécification des langages informatiques. Ce domaine est pratiquement aussi ancien que l'informatique elle-même et se confond avec la création des premiers langages de programmation et des premiers compilateurs. Ce domaine a pris un essor important avec le développement des langages et modèles spécifiques aux domaines, connus sous les sigles DSL et DSM (Sprinkle *et al.*, 2009). Le second courant est celui des outils et environnements logiciels de méta-modélisation. Certains de ces outils ont pour objectif explicite la définition de nouveaux langages, notamment des DSL et DSM, tel que MetaEdit+ par exemple (Kelly et Tolvanen, 2008) ; d'autres font partie d'ateliers sophistiqués de génie logiciel tel que TOPCASED (Farail, 2012). Dans cette section, nous faisons une brève synthèse de ces deux courants de recherche en s'inspirant des présentations faites dans (Gargantini *et al*, 2009) et (Bryant *et al.*, 2011). Notre objectif est de faire le point sur la manière avec laquelle la spécification de la sémantique est faite, est-elle déclarative, est-il possible de la représenter graphiquement ; et si éventuellement, elle permet la génération automatique d'un outil d'exécution.

La technique la plus connue pour définir la sémantique d'un langage de programmation est la grammaire des attributs introduite par D. Knuth en 1968. A chaque nœud de la syntaxe abstraite d'un langage est associé un (ou plusieurs) attribut(s), ainsi que des fonctions pour calculer et propager la valeur de ces attributs à partir des attributs des nœuds adjacents. Cette technique a été largement appliquée en compilation pour construire des analyseurs sémantiques et des générateurs de code. Le concepteur décrit ainsi sous forme d'opérations et de calculs la sémantique des constructions du langage. Cette vision *opérationnelle* et *impérative* de la sémantique se distingue d'autres approches déclaratives et plus formelles, telles que les sémantiques axiomatique et dénotationnelle (Winskel, 1993).

Généralement associée avec le développement des langages à syntaxe textuelle, la grammaire des attributs a été adaptée à la spécification des langages de modélisation. Le prototype de recherche JastEMF (Bürger *et al.*, 2011) combine l'environnement de méta-modélisation EMF d'Eclipse avec le méta-compilateur expérimental JastAdd basé sur la grammaire des attributs (Hedin, 2011). JastEMF est de ce fait un environnement de méta-modélisation qui intègre la puissance de la grammaire des attributs. L'expression de la sémantique est en partie déclarative (fonctions rattachées aux noeuds) et en partie impérative (des calculs dans les fonctions). Cependant, elle est textuelle et n'a pas de représentation graphique.

Dans l'univers de l'IDM, le langage Kermeta est une des approches les plus innovantes. Dans (Jézéquel *et al.*, 2011), les auteurs définissent Kermeta comme un méta-atelier pour *l'Ingénierie des Langages Dirigée par les Modèles*. C'est une approche qui se situe dans l'univers du méta-méta-modèle MOF (Meta Object Facility). Kermeta utilise Ecore, une variante du MOF intégrée avec l'environnement Eclipse, pour définir la syntaxe abstraite d'un langage sous forme d'un méta-modèle statique. La sémantique statique (contraintes sur le méta-modèle) s'exprime avec le langage OCL, et la sémantique d'exécution avec des méthodes directement rattachées aux méta-classes à l'aide d'un langage de programmation spécifique orienté aspects inspiré de Java. A partir de cette méta-spécification, Kermeta génère un éditeur, un vérificateur et un exécuteur pour le langage en cours de construction. Cette spécification de la sémantique opérationnelle n'est pas déclarative et ne possède pas de représentation graphique, aucune conceptualisation n'étant disponible pour décrire les séquences d'instructions.

Un autre projet très abouti dans le monde de l'IDM est TOPCASED. TOPCASED est un atelier de génie logiciel destiné au développement de systèmes embarqués (Farail, 2012). C'est une solution basée sur la plateforme logicielle Eclipse et utilise les langages UML et SysML. TOPCASED est aussi un méta-environnement pour la définition de nouveaux langages (Crégut *et al.*, 2010). Comme dans Kermeta, la syntaxe abstraite s'exprime avec Ecore et elle est complétée par une syntaxe concrète définie avec un éditeur graphique. La sémantique du méta-modèle s'exprime à l'aide de formalismes à base d'état et d'événements. Cette spécification permet la simulation d'exécution à l'aide de trois éléments : *l'Agenda*, *le Contrôleur* (ou Driver), et *l'Interpréteur*. Elle est en partie déclarative, graphique et formelle

grâce aux méta-modèles d'états et d'événements. Deux éléments du moteur d'animation sont génériques, alors que l'interpréteur est réécrit pour chaque langage et cette écriture n'est ni déclarative ni graphique. Dans (Crégut *et al.*, 2010), les auteurs proposent l'identification de motifs génériques (*patterns*) pour y remédier.

Enfin, dans le monde DSL et DSM, un des ateliers les plus connus est MetaEdit+ (Kelly et Tolvanen, 2008)(MetaCASE, 2012). C'est un outil de méta-modélisation, de génération d'ateliers de génie logiciel et d'ingénierie des méthodes qui est régulièrement cité et évalué (Niknafs et Ramsin, 2008)(El Kouhen *et al.*, 2012). La spécification de la sémantique se fait au niveau des scripts de génération de code, elle n'est ni déclarative ni graphique. Et nous avons pu constater dans un projet exploratoire que nous avons mené (Mallouli et Assar, 2013), que les fonctionnalités de méta-modélisation sont effectivement puissantes et faciles à utiliser. Cependant, la programmation du générateur de code est rendue très complexe par l'absence de représentation graphique de l'architecture du générateur.

Ce bref tour d'horizon de quelques travaux relatifs à l'ingénierie des langages nous permet de dresser le constat suivant : la méta-modélisation est une technique qui est actuellement supportée par des outils sophistiqués et fiables, cependant, lorsqu'il s'agit de spécifier un langage de modélisation ayant une sémantique exécutable, l'intégration de la spécification de la sémantique dans le méta-modèle reste difficile. Plusieurs approches sont explorées dans la littérature, elles restent pour le moment difficiles à mettre en œuvre. Pour notre part, nous considérons qu'il est important que cette spécification de la sémantique puisse être représentée graphiquement et qu'elle soit exprimée dans une logique déclarative et non pas uniquement opérationnelle. C'est l'objectif de la démarche que nous présentons ici.

3. Méta-modélisation événementielle

L'étude de l'état de l'art illustre la diversité des approches pour spécifier le plus précisément possible la sémantique d'un langage, et surtout, pour exploiter cette spécification dans la construction des outils nécessaires à l'utilisation concrète du langage. Pour élaborer notre approche, nous avons cherché une spécification de la sémantique d'un langage qui satisfait plusieurs contraintes : (i) Elle doit être de nature **déclarative** pour faciliter son élaboration; (ii) elle doit avoir une représentation **graphique** qui facilite sa lecture et sa compréhension; (iii) elle doit posséder une **sémantique claire** pour éviter les ambiguïtés d'interprétation ; et enfin, (iv) cette spécification doit être suffisamment **riche** pour qu'il soit possible d'en déduire, par génération de code, un outil d'exécution de modèles.

Nous avons choisi d'élaborer notre approche selon une logique d'IDM. En effet, notre objectif étant de construire des outils logiciels (pour exécuter des modèles de processus), il nous parait naturel de favoriser l'usage de modèles et d'exploiter ainsi la démarche IDM pour d'une part, offrir au concepteur de langage des spécifications de niveau conceptuel et d'autre part, obtenir des spécifications de niveau physique pour réaliser l'outil logiciel. Le synopsis de cette approche est présenté à la figure 1.

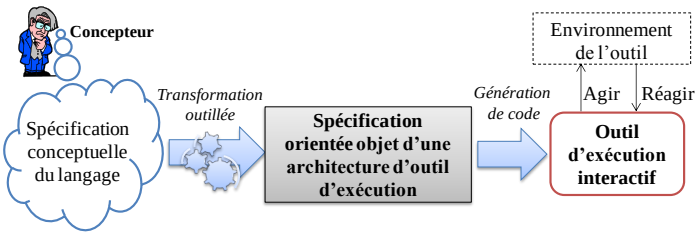


Figure 1. *Synopsis de notre approche*

La spécification conceptuelle du langage comporte deux parties selon deux visions spécifiques et complémentaires : un diagramme de classes UML pour une vision structurelle du méta-modèle, et un schéma événementiel pour une vision dynamique de la sémantique d'exécution du langage. Le paradigme événementiel est très utilisé pour décrire les systèmes réactifs et asynchrones, et nous semble très approprié pour capturer la vue systémique d'un outil d'exécution. Il n'est cependant pas présenté comme un concept central des langages tels qu'UML ou réseaux de Pétri. Ceci nous a amené à choisir le formalisme dynamique de la méthode Remora (Rolland *et al*, 1988) pour sa concision (peu de concepts), son expressivité pour la vue systémique (événement et échange de messages), sa sémantique claire et bien définie ainsi que les possibilités d'implémentation.

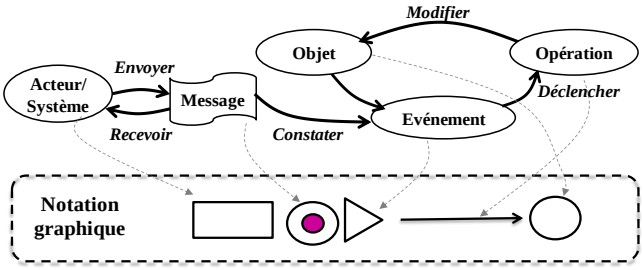


Figure 2. *Concepts et notation graphique du formalisme de modélisation Remora*

La figure 2 résume les concepts de Remora ainsi que la notation graphique. On note la présence du concept « acteur » qui peut être un agent humain ou un système (ou application) externe. Ce concept sert à décrire les entités qui se trouvent dans l'environnement du système et avec lesquels il échange des messages (entrants et sortants). L'envoi et la réception de ces messages constituent des événements externes qui agissent sur la dynamique du système en déclenchant des opérations.

La figure 3 décrit en détail notre approche IDM pour la spécification d'un langage de modélisation. Dans cette approche, la spécification de la sémantique d'exécution – capturée par le schéma événementiel – correspond en fait à la logique opérationnelle d'exécution d'un outil logiciel qui interprèterait les instances du modèle. C'est une vision opérationnelle de la sémantique, elle revient à spécifier le fonctionnement d'un interpréteur abstrait du modèle.

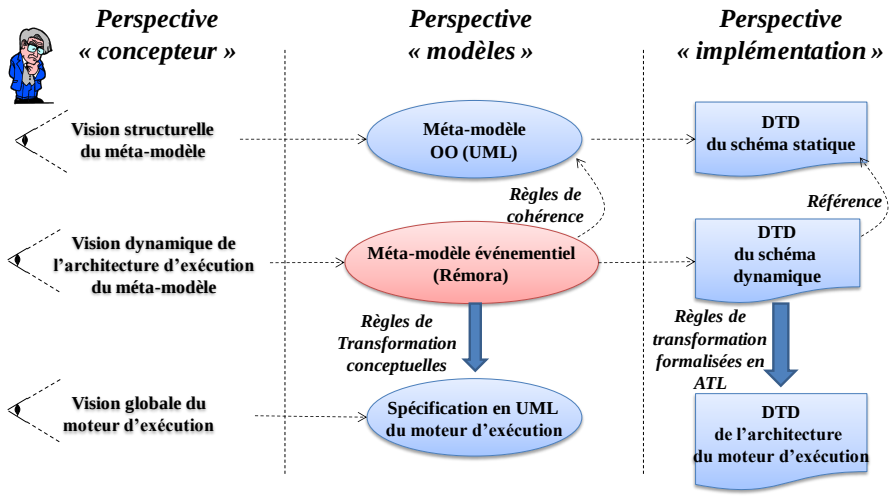


Figure 3. *Vue détaillée de notre approche selon trois perspectives*

Dans la première étape, le concepteur définit le méta-modèle structurel (i.e. la syntaxe abstraite) du langage sous forme d'un diagramme de classes UML. A l'étape suivante, le concepteur traduit la sémantique du langage à l'aide d'un schéma événementiel Remora. Comme ce schéma se confond avec la logique opérationnelle de fonctionnement d'un interpréteur, il faut au préalable rajouter aux méta-classes du diagramme UML un ensemble de classes pour représenter **les instances des concepts** du modèle. Ainsi, à chaque méta-classe est définie par extension une classe d'instances. Cet ensemble de classes et de méta-classes, organisé en deux niveaux d'abstractions, contient les objets sur lesquels va porter le schéma événementiel. Dans la perspective « implémentation », ces diagrammes de classes UML seront décrits dans des documents XML (fichiers avec extension .DTD).

4. Dérivation de l'architecture d'un outil d'exécution

La troisième étape de la démarche consiste à générer l'architecture technique de l'outil d'exécution de modèles. Il s'agit de traduire la sémantique d'exécution du modèle capturée dans le schéma événementiel en une spécification logicielle complète. Pour traduire la logique événementielle d'un schéma Remora, nous faisons appel à un cadre (ou « framework ») d'exécution connu, celui des motifs génériques « publier/souscrire » (Eugster *et al.*, 2003). Le résultat de cette transformation est une architecture logicielle décrite dans un diagramme de classes UML complet qui englobe les méta-classes, les classes d'extension ainsi que les classes qui implémentent la logique opérationnelle de l'outil d'exécution.

4.1 Le framework d'exécution asynchrone publier/souscrire

Les motifs génériques « publier/souscrire » sont issus de l'univers de la programmation distribuée, et sont considérés comme le paradigme de choix pour le développement d'applications asynchrones et réactives (Hinze *et al.*, 2009). Le principe fondamental est le suivant: un objet (appelé « objet dynamique ») qui doit réagir à l'occurrence d'un événement constaté sur un objet « observé » va souscrire à cet événement. A chaque occurrence de celui-ci, le système se chargera de le propager aux objets abonnés en les notifiant de l'occurrence de l'événement et en leur communiquant les données contextuelles associées. L'objectif est de permettre à un ou plusieurs objets de réagir aux messages d'autres objets, sans qu'ils ne soient connus à l'avance, sans devoir les lier « en dur » dans le code. Pour la transformation d'un schéma Remora, nous faisons appel à trois motifs d'exécution. Le premier correspond à une interaction interne entre objets appartenant à un même système. C'est une implémentation directe du paradigme « publier/souscrire », elle est schématisée dans la figure 4.

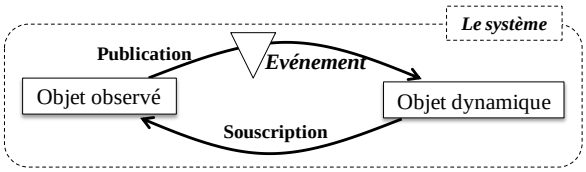


Figure 4. *Motif pour la gestion d'une interaction interne*

Le second motif correspond à une interaction externe, où un message entant venant de l'extérieur du système (une application externe) est attendu par des objets dynamiques. Ce motif est représenté par une interaction entre l'objet dynamique et une file d'attente dans laquelle sont déposés les messages de l'objet observé, et à laquelle va souscrire l'objet dynamique (figure 5).

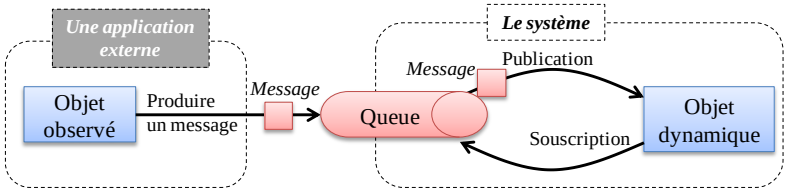


Figure 5. *Motif pour la gestion d'une interaction externe avec message entrant*

Le troisième motif correspond aussi à une interaction externe dans laquelle le message est produit par un objet dynamique du système pour être consommé par une application externe particulière. C'est le cas par exemple d'une invocation d'une application externe pour exécuter une tâche du processus, ou le cas d'une notification d'un état du système à destination d'un acteur externe. C'est une interaction de type « point à point » où le message produit par l'objet observé est directement consommé – après passage dans une file d'attente de type FIFO – par un seul objet consommateur (figure 6).

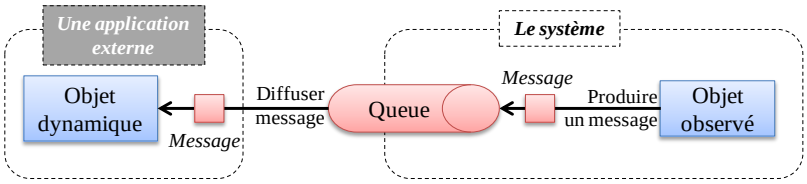


Figure 6. *Motif pour la gestion d'une interaction externe avec message sortant*

4.2 Les règles de transformation

Nous allons maintenant exploiter ces trois motifs d'exécution pour transformer le schéma dynamique Remora en une architecture logicielle. Une première analyse des concepts du formalisme source (figure 2) et celles du formalisme cible (diagramme de classes UML) permet d'identifier plusieurs transformations directes :

• Les éléments *Objet* n'ont pas à être transformés, elles correspondent aux classes-instances et méta-classes du diagramme UML.

• Lorsqu'elle modifie un objet, une *Opération* se transforme en une méthode ; sinon, elle est traitée par une règle

• Un lien *Modifie* entre un *Objet* et *Opération* se transforme en un lien d'appartenance de la méthode à la classe correspondant à l'*objet.*

• Un lien *Déclenche* entre *Événement* et *Objet* est transformé en une méthode d'une classe particulière appelée *Listener*. Cette méthode est invoquée après la constatation d'un événement (interne ou externe) qui lui est donné en paramètre. Le type du *Listener* dépend de l'événement (s'il est interne ou externe).

Au-delà de ces règles directes, nous introduisons trois groupes de règles pour la transformation des événements internes, des événements externes et des opérations qui communiquent avec les acteurs externes.

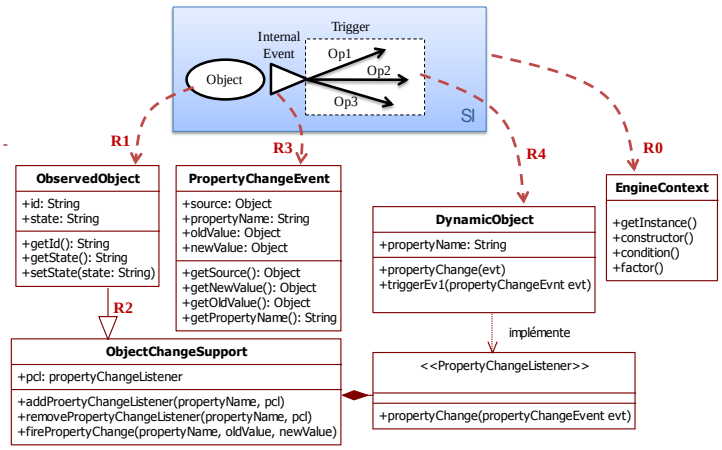


Figure 7. *Transformation d'un événement interne*

Le premier groupe des règles traduit un événement interne en une structure d'interaction interne de type « publier/souscrire » (cf. fig. 4). Ce qui correspond à cinq règles illustrées à la figure 7 :

R0. Une classe *EngineContext* est créée pour gérer les données génériques d'exécution des processus (niveau des concepts) et des données spécifiques du processus en cours d'exécution (niveau des instances).

R1. A la classe qui correspond à l'objet sur lequel est constaté l'événement, on crée une classe *ObservedObject* avec une propriété d'état et des méthodes prédéfinies *getState()* et *setState()* pour y accéder.

R2. La classe *ObservedObject* hérite de la classe prédéfinie *ObjectChangeSupport* afin d'avoir des capacités de gestion d'objets dynamiques qui s'abonnent à des événements. Cette super classe permet de créer ou de supprimer de nouveaux *Listener* correspondant à de nouveaux abonnements d'objets dynamiques. La méthode *firePropertyChange()* permet de notifier le changement d'état d'une propriété (événement), de générer une instance de l'objet *PropertyChangeEvent* et de diffuser cet événement aux abonnés à cette propriété en invoquant la méthode *propertyChange()*. Pour que ce mécanisme soit générique, les objets abonnés doivent être issus d'une classe qui implémente l'interface *PropertyChangeListener*.

R3. La classe prédéfinie *PropertyChangeEvent* représente de manière générique tous les événements internes du schéma dynamique. Un événement de type *PropertyChangeEvent* est défini par un nom de propriété (*propertyName*), l'ancienne et la nouvelle valeur (*oldValue, newValue*) et une référence de l'objet sur lequel l'événement est constaté (*source*).

R4. Associer au groupe d'opérations déclenchées par un événement interne une classe *DynamicObject* (observateur de l'événement). Cette classe implémente l'interface <<*PropertyChangeListener*>> et contient ainsi la méthode *propertyChange()* qui prend comme paramètre l'événement déclenchant. Dans cette méthode s'effectue l'acheminement des opérations du déclencheur en appliquant des tests sur la structure de l'événement qui est passé en paramètre.

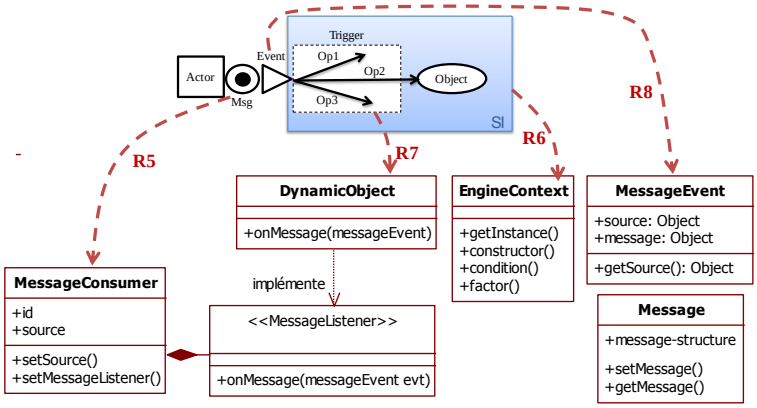


Figure 8. *Transformation d'un événement externe avec un message entrant*

Le second groupe de règles concerne les événements externes. En appliquant le motif d'interaction externe avec un message entrant (fig. 5), un acteur est considéré comme un producteur de messages. On souscrit alors les objets sur lesquels portent les opérations déclenchées par l'événement externe aux messages publiés par cet acteur. Ce qui se traduit par les règles illustrées à la fig.8.

R5. Pour l'événement externe, définir une file d'attente à travers la classe *MessageConsumer* qui comprend l'interface <<*MessageListener*>>. Cette interface souscrit à la file d'attente *MessageConsumer* et la méthode *onMessage()* est invoquée lorsqu'il y a un nouveau message à consommer dans la file d'attente. L'acteur lui-même apparaîtra dans la classe *MessageConsumer* via l'attribut *source*.

R6. Pour chaque groupe d'opérations déclenchées par un événement, pour chaque condition et chaque facteur, ajouter les méthodes correspondantes dans la classe globale *EngineContext*.

R7. Associer au groupe d'opérations déclenchées par un événement externe une classe *DynamicObject* qui implémente l'interface <<*MessageListener*>>. Cette classe souscrira à la file d'attente correspondante à l'événement externe. A l'arrivée d'un message, l'objet *MessageConsumer* génère une instance de *MessageEvent* avant d'invoquer la méthode *onMessage()* de la classe DynamicObject.

R8. Associer à un message une classe *MessageEvent*, ce message est consommé par la méthode *onMessage()* de l'objet dynamique.

Le 3ème groupe de règles transforme une opération qui invoque ou notifie un acteur externe en son équivalent en termes de concepts UML orientés objet. Ceci est réalisé avec le motif pour la gestion d'une interaction externe avec message sortant (cf. fig.6). Dans cette situation, l'objet est le producteur du message et l'acteur externe en est le consommateur. Ce qui se traduit par les règles illustrées à la fig.9.

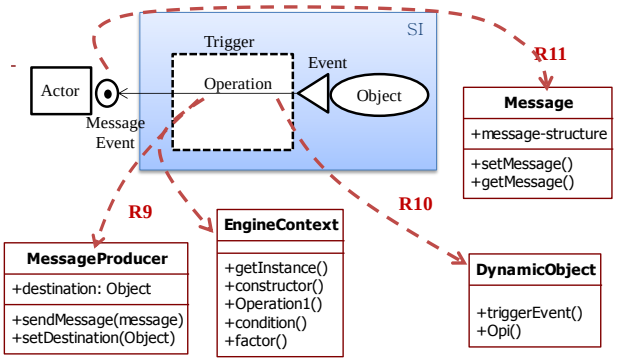


Figure 9. *Transformation d'une opération externe*

R9. Associer à l'opération externe une classe *MessageProducer*. Ajouter à cette classe une méthode *sendMessage()* pour diffuser le message. Référencer l'objet dynamique destinataire du message dans la classe *MessageProducer* avec l'attribut *destination*. La création d'une instance de *MessageProducer* est réalisée par la méthode *constructor*() de la classe globale *EngineContexte*.

R10. Pour l'opération déclenchée, ajouter l'opération externe *Opi* en tant que méthode dans l'objet *DynamicObject*.

R11. Le message envoyé par l'opération externe est définie dans une classe *Messsage* et il est géré grâce aux méthodes *getMessage()* et *setMessage().*

5. Illustration

Nous avons expérimenté notre approche sur le langage Map de modélisation intentionnel de processus (Rolland *et al.*, 1999). Ce langage est particulièrement adapté pour représenter des processus à haut niveau d'abstraction et à forte variabilité. La sémantique complexe d'exécution du Map correspond à une navigation dans un graphe selon les intentions de l'utilisateur et suivant le contexte situationnel relatif à l'état du produit. C'est un excellent exemple pour illustrer l'applicabilité de notre approche et son intérêt pour construire un outil d'exécution.

Brièvement, une carte Map est un ordonnancement non figé d'intentions reliées par des stratégies. Une stratégie est une manière de réaliser une intention. Elle peut être une action exécutée par une application externe et qui aboutit à des transformations du produit, ou être elle-même décrite récursivement par une sous-carte Map. Ce qui autorise une grande variabilité dans l'ordonnancement de réalisation des intentions et dans le choix des stratégies à appliquer.

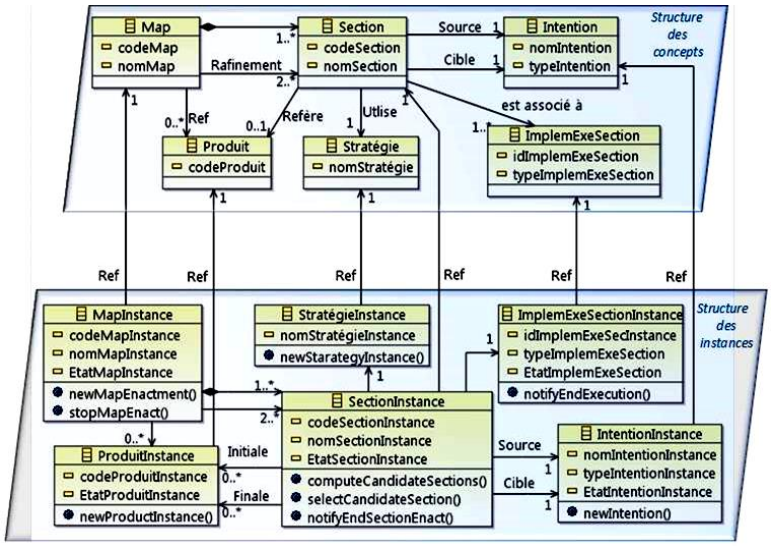


Figure 10. *Modèle structurel à deux niveaux (concept-type et instances) du Map*

La figure 10 présente le résultat de l'application de la 1ère étape. Le méta-modèle statique s'organise autour du concept de section. Une section représente un triplet constitué d'une Intention source, une Intention cible et une Stratégie. Cet élément est associé au concept abstrait *ImplemExeSection* qui renseigne sur la manière

d'exécuter une section (service web, ensemble de directive ou application externe). Une section est aussi reliée à un produit sur lequel porte l'exécution de la section.

Pour opérationnaliser cette structure statique, il faut rajouter des structures d'instances sous forme d'un ensemble de classes-instances (cf. section 4). Ensuite, en se basant sur la sémantique d'exécution du modèle Map, on complète par des méthodes et des attributs d'états nécessaires pour l'exécution du moteur du Map. Ces ajouts ne se font pas forcément dans cette étape, mais peuvent l'être lors de la spécification du schéma dynamique qui exprime la sémantique d'exécution du Map à l'aide d'un schéma événementiel Remora (fig.11). Dans ce schéma, on considère chaque classe du modèle structurel comme un objet susceptible d'avoir des changements d'états et sur lequel peuvent être constatés des événements.

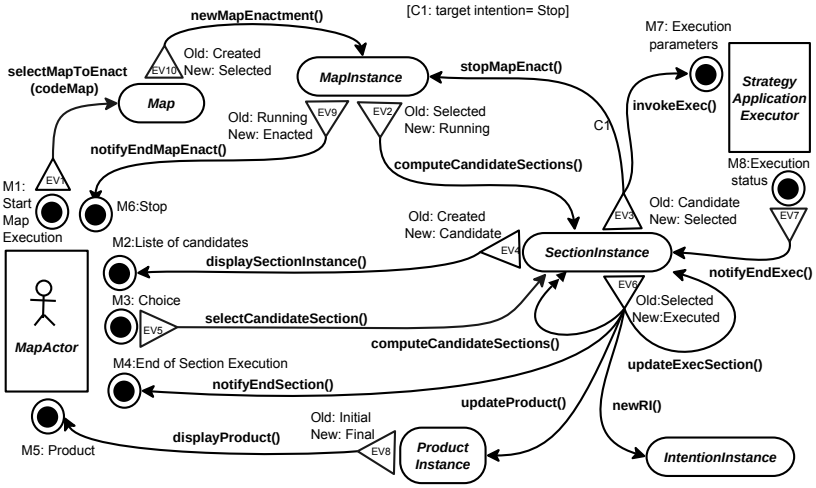


Figure 11. Spécification événementielle de la sémantique d'exécution du Map

Nous définissons par ailleurs deux acteurs externes : *MapActor* pour représenter l'utilisateur final qui exécute une carte, et *StrategyApplicationExecutor* pour un agent externe qui exécute les actions atomiques associées aux stratégies.

6. Évaluation

Le modèle des cartes de processus Map avec lequel nous avons illustré notre démarche a fait l'objet de plusieurs travaux de recherche visant à construire un outil d'exécution. Pour situer l'apport et la pertinence de cette démarche, nous la comparons avec ces travaux antérieurs selon trois catégories de critères (tableau 1) : la démarche, la méta-modélisation et les caractéristiques de l'outil d'exécution final.

La démarche que nous proposons ici est, d'un point de vue méthodologique, beaucoup plus structurée que les autres approches. Les travaux de (Velez, 2003) et (Edme, 2005) adoptent une démarche spécifique, ad-hoc et non structurée, directement orientée vers la production d'un outil logiciel d'exécution de carte Map.

Du point de vue de la méta-modélisation, ces deux travaux utilisent certes un méta-modèle des cartes Map dans leur raisonnement, mais la sémantique d'exécution est directement programmée dans l'outil.

		Approches			
Critères		**(Velez'03)**	**(Edme'05)**	**MétaEdit+** (Mallouli et al.' 13)	**Notre proposition**
Démarche	Nature	Ad-hoc	Ad-hoc	Meta-CASE	IDM
	Générique/ spécifique	Spécifique	Spécifique	Générique	Générique
Méta-modélisation	Formalisme statique	E/R	E/R	GOPRR	MOF (UML)
	Expression de la sémantique d'exécution	100% code orienté objet	Requête SQL + code VB + tables relationnelles	Script MERL + code généré	Schéma événementiel de l'architecture du moteur
Outil d'exécution	Interactivité	Non	Non	Non	Oui
	Maintenabilité	-	-	+	++
	Portabilité	-	-	+	+

Tableau 1. *Évaluation de notre démarche par rapport à des approches similaires*

La démarche exploratoire expérimentée dans (Mallouli et Assar, 2013) est plus proche de celle développée ici. Elle repose certes sur l'usage de méta-modèles explicites grâce à l'utilisation du Meta-CASE (i.e. un outil de méta-modélisation) MetaEdit+ ; néanmoins, la sémantique d'exécution s'exprime dans un ensemble de scripts de génération de code, sans représentation graphique de cette sémantique. C'est suite à ce projet exploratoire que nous avons été convaincus de l'apport potentiel d'une approche IDM pour construire des outils d'exécution de modèles.

Enfin, du point de vue de l'outil logiciel obtenu, les démarches de programmation directes de (Velez, 2003) et (Edme, 2005) aboutissent à des prototypes très difficiles à maintenir et à porter sans redéveloppement complet. La démarche par méta-CASE de (Mallouli et al., 2013) aboutit à une solution plus intéressante, mais en cas d'évolution du langage, elle nécessite une mise à jour du code généré et des scripts de génération. Avec la démarche présentée ici, l'évolution du langage de modélisation nécessite une révision des méta-modèles statiques et dynamiques, et une nouvelle application des règles de transformations pour obtenir un nouvel outil d'exécution. La mise à jour d'un schéma graphique est généralement plus aisée que celle d'un ensemble de lignes de code, fut-ce des scripts de génération de code comme c'est le cas avec MetaEdit+.

7. Conclusion

Lorsqu'on développe un nouveau langage de programmation ou de modélisation, la construction d'outils logiciels pour le supporter se pose rapidement. Les approches de type méta-programmation et méta-modélisation se heurtent à la question ardue de l'expression de la sémantique du langage, surtout si cette

sémantique est exécutable comme c'est le cas avec les langages de modélisation de processus. Par rapport aux approches existantes, nous avons proposé dans cet article une démarche de type IDM qui introduit une spécification graphique et rigoureuse de la sémantique d'exécution du langage. Cette spécification est de nature opérationnelle, elle traduit le fonctionnement d'un outil qui interpréterait les instances de ces modèles, et nous avons défini des règles de transformation pour générer l'architecture logicielle d'un outil d'exécution. La validation de cette démarche est en cours, nous travaillons actuellement à l'implémentation des règles de transformation avec le langage ATL dans le cadre d'un prototype. Néanmoins, une première application de cette approche a permis de mettre en évidence son intérêt par rapport à des démarches plus classiques expérimentées dans des travaux antérieures. La validité externe est pour le moment un peu limitée, elle nécessiterait l'application de la méthode sur d'autres langages et une comparaison plus aboutie avec des approches concurrentes. C'est vers cet horizon que tendent nos travaux futurs.

8. Bibliographie

Atkinson, C., Kühne, T., "Model-Driven Development: A Metamodeling Foundation", *IEEE Software*, 20(5), p. 36–41, 2003.

Bryant, B., Gray, J., Mernik, M., Clarke, P., France, R., Karsai, G., "Challenges and Directions in Formalizing the Semantics of Modeling Languages", *Computer Science and Information Systems*, 8(2), p. 225‑253, 2011.

Bürger, C., Karol, S., Wende, C., Aßmann, U., "Reference Attribute Grammars for Metamodel Semantics", dans B. Malloy, S. Staab, M. van den Brand (éds), *Software Language Engineering*, Vol. 6563, p. 22‑41, Springer, 2011.

Crégut, X., Combemale, B., Pantel, M., Faudoux, R., Pavei, J., "Generative Technologies for Model Animation in the TopCased Platform" dans T. Kühne, B. Selic, M.-P. Gervais, F. Terrier (éds.), *Modelling Foundations and Applications*, p. 90‑103, Springer, 2010.

Edme, M., Proposition pour la modélisation intentionnelle et le guidage de l'usage des systèmes d'information (Thèse de doctorat). Univ. Paris 1 La Sorbonne, France, 2005.

El Kouhen, A., Dumoulin, C., Gerard, S., Boulet, P., "Evaluation of Modeling Tools Adaptation", 2012, consulté à http://hal.archives-ouvertes.fr/hal-00706701.

Eugster, P. T., Felber, P. A., Guerraoui, R., Kermarrec, A.-M., "The many faces of publish/subscribe", *ACM Comp. Surveys*, 35(2), p. 114–131, 2003.

Farail, P., "Toolkit in OPen-source for Critical Applications & SystEms Development", 2012, consulté à http://www.topcased.org/.

Favre, J.-M., Estublier, J., Blay-Fornarino, M., *L'ingénierie dirigée par les modèles au-delà du MDA*, Paris, France: Lavoisier–Hermès Sciences, 2006.

Favre, J.-M., Gasević, D., Lammel, R., Winter, A., "Guest Editors' Introduction to the Special Section on Software Language Engineering", *IEEE Transactions on Software Engineering*, 35(6), p. 737‑741, 2009.

Gargantini, A., Riccobene, E., Scandurra, P., "A semantic framework for metamodel-based languages", *Automated Software Engineering*, *16*(3-4), p. 415‑454, 2009.

Harel, D., Rumpe, B., "Meaningful modeling: what's the semantics of « semantics »?", *IEEE Computer*, 37(10), p. 64‑72, 2004.

Hedin, G., "An Introductory Tutorial on JastAdd Attribute Grammars", dans J. Fernandes, R. Lämmel, J. Visser, J. Saraiva (éds.), *Generative and Transformational Techniques in Software Engineering III*, Vol. 6491, p. 166‑200, Springer, 2011.

Hinze, A., Sachs, K., Buchmann, A., "Event-based applications and enabling technologies, dans *Proc. 3rd ACM Int. Conf. on Distributed Event-Based Systems*, p. 1–15, ACM, 2009.

Jézéquel, J-M., Barais, O., Fleurey, F., "Model Driven Language Engineering with Kermeta", dans J. M. Fernandes, R. Lämmel, J. Visser, J. Saraiva (éds.), *Generative and Transformational Techniques in Software Engineering III*, p. 201‑221, Springer, 2011.

Jézéquel, J.-M., Combemale, B., Derrien, S., et al. "Bridging the chasm between MDE and the world of compilation", *Software & Systems Modeling*, 11(4), p. 581‑597, 2012.

Jouault, F., Bézivin, J., Barbero, M., "Towards an advanced model-driven engineering toolbox", *Innovations in Systems and Software Engineering*, 5(1), p. 5‑12, 2009.

Kelly, S., Tolvanen, J-P., *Domain-specific modeling: enabling full code generation*. Hoboken, N.J.: Wiley-Interscience: IEEE Computer Society, 2008.

Kleppe, A., *Software language engineering: creating domain-specific languages using metamodels*, Addison-Wesley Professional, 2009a.

Kleppe, A., "The Field of Software Language Engineering", dans D. Gasevic, R. Lämmel, E. V. Wyk (éds.), *Software Language Engineering (SLE'08)*, p. 1‑7, Springer, 2009b.

Knuth, D. E., "Semantics of context-free languages", *Theory of Computing Systems*, 2(2), p. 127–145, 1968.

Mallouli, S., Assar, S., "Enacting a Requirement Engineering Process with Meta-Tools: an Exploratory Project" *Proceedings 8th Int. Multi-Conf. on Computing in the Global Information Technology (ICCGI 2013)*, p. 208‑213, 2013.

MetaCASE, consulté à http://www.metacase.com/, 2012.

Niknafs, A., Ramsin, R., "Computer-Aided Method Engineering: An Analysis of Existing Environments" dans Z. Bellahsène et al. (éds.), *CAiSE'08*, p. 525‑540, Springer, 2008.

Paakki, J., "Attribute grammar paradigms—a high-level methodology in language implementation", *ACM Computing Surveys*, 27(2), p. 196–255, 1995.

Rolland, C., Foucault, O., Benci, G., *Conception des systèmes d'information: la méthode REMORA*, Paris, France: Eyrolles, 1988.

Rolland, C., Prakash, N., Benjamen, A., "A Multi-Model View of Process Modelling", *Requirements Engineering*, 4(1), p. 169‑187, 1999.

Sprinkle, J., Mernik, M., Tolvanen, J., Spinellis, D., "Guest Editors' Introduction: What Kinds of Nails Need a Domain-Specific Hammer?", *IEEE Software*, 26(4), 15‑18, 2009.

Sprinkle, J., Rumpe, B., Vangheluwe, H., Karsai, G., "Metamodelling: State of the Art and Research Challenges", dans H. Giese, G. Karsai, E. Lee, B. Rumpe, B. Schatz (éds.), *Model-Based Engineering of Embedded Real-Time Systems*, p. 57‑76, Springer, 2011.

Velez, F., *Proposition d'un environnement logiciel centré processus pour l'ingénierie des systèmes d'information* (Thèse de doctorat). Univ. Paris 1 La Sorbonne, France, 2003.

Winskel, G., *The Formal Semantics of Programming Languages*, MIT Press, 1993.

Session 4a

Manipulation, visualisation et exploitation de modèles complexes

Des situations de modélisation pour évaluer les outils de modélisation

Antoine Beugnard* — **Fabien Dagnat*** — **Sylvain Guérin**** — **Christophe Guychard****

* *Télécom Bretagne, IRISA*
Technopole Brest-Iroise, CS 83818, 29283 Brest cedex 3, France
prenom.nom@telecom-bretagne.eu
** *Openflexo*
Technopole Brest-Iroise, 135 rue Claude Chappe, 29280 Plouzané, France
prenom.nom@openflexo.org

RÉSUMÉ. *Nous proposons d'identifier des situations de modélisation en mettant en évidence des actions élémentaires sur les artéfacts de modélisation que sont les modèles et les méta-modèles. Nous pensons que l'identification de ces situations permettra de mieux comprendre la modélisation et par conséquent les besoins des outils de modélisation. Nous présentons Openflexo, un outil de* modélisation libre*, et esquissons une comparaison de ses capacités avec divers type d'outils de modélisation qui vont du simple outil dessin à l'atelier de méta-modélisation.*

ABSTRACT. *We propose to identify modeling situations highlighting elementary actions on modeling artifacts such as models and meta-models. We believe that identification of these situations will help better understand the modeling and therefore modeling tools requirements. We present Openflexo a* free modeling *tool, and sketch a comparison of its capabilities with various types of modeling tools ranging from simple drawing tool to meta-modeling editors.*

MOTS-CLÉS : *Modélisation*

KEYWORDS: *Modeling*

1. Introduction

La modélisation est à la base de la plupart des stratégies de résolution de problèmes, en particulier de la démarche scientifique et de l'ingénierie (Morris, 1967 ; Lachapelle et Cunningham, 2007). On modélise en s'appuyant sur des écrits[1] sous formes de phrases ou de dessins. Ces phrases et dessins, artéfacts issus de la modélisation, doivent respecter des règles de construction plus ou moins explicites ou formalisées.

Dans « On the art of modeling » (Morris, 1967), W. Morris propose de passer d'un processus de modélisation intuitif à une approche explicite. Il illustre l'article par un problème de planification de transport. Au-delà des étapes de réflexion, il fait apparaître deux étapes qu'on retrouve dans tout processus de modélisation :

1) *Consider a specific (. . .) instance of the problem* ; identifier des exemples.

2) *Establish some symbols* ; déterminer leur généralisation ou abstraction, et en définir des représentations, à travers des variables mathématiques par exemple.

Il note que la production de ces artéfacts (les exemples et les symboles) suit un processus d'élaboration par enrichissement :

> « The process of model development may be usefully viewed as a process of *enrichment* or *elaboration*. One begins with very simple models, quite distinct from reality, and attempts to move in evolutionary fashion toward more elaborate models which more nearly reflect the complexity of the actual management situation. »

Enfin, il met également en évidence le besoin de liens (implicites ou explicites) entre ces artéfacts :

> « *Analogy* or *association* with previously well developed logical structures plays an important role in the determination of the starting point of this process of elaboration or enrichment. »

Dans la suite de cet article, nous appelons *modèle* une représentation explicite d'un système ou d'un problème et *méta-modèle*, l'ensemble des règles explicites - quel que soit leur mode d'explicitation - qu'un modèle doit respecter. Nous verrons que les liens entre modèle et méta-modèle peuvent être connu *a priori*, mais sont parfois élaborés *a posteriori*, comme le résultat d'un choix de modélisation.

La formalisation des outils de modélisation s'est concrétisée, pour les phrases, par la théorie des langages et des grammaires, pour les dessins, par les approches dites dirigées par les modèles. Dans les deux cas, les modèles produits (phrases ou dessins) sont *conformes* aux règles exprimées respectivement dans la grammaire ou le

1. Même si les premiers philosophes discutaient sur l'Agora sans laisser de traces écrites et que nos ancêtres ont dû résoudre bien des problèmes avant l'invention de l'écriture. . .

méta-modèle. Cette relation de conformité est l'une des relations possibles entre modèles et méta-modèles. Nous considérerons donc deux niveaux de modélisation : le modèle (exemple, instance, concrétisation) et le méta-modèle (généralisation, abstraction). Dans le sens où nous l'utilisons, le méta-modèle (Kleppe, 2007) est l'expression des concepts (lexique, ou vocabulaire) utilisables pour élaborer un modèle et de toutes les règles et contraintes (grammaire) d'assemblage de ces concepts. Notre approche peut être considérée comme « catégorique » au sens mathématique du terme. Nous nous intéressons aux relations entre modèles et méta-modèles sans prendre en considération leur structure ou leur contenu.

Nous noterons que dans la plupart des outils et démarches de modélisation existants, la relation de conformité est considérée comme stricte i.e. l'élément modélisé conserve exactement et uniquement les propriétés définies dans son méta-modèle, mais nous ne retenons pas cette hypothèse dans notre démarche. En effet, elle se révèle parfois être une contrainte prescriptive qui appauvrit l'expressivité des modèles. Enfin, on rencontre aussi souvent la relation d'instantiation (*instanceof*) qui relie un modèle créé depuis un méta-modèle à ce méta-modèle. Dans le cadre de cet article et suivant la vision catégorique déjà exposée, nous ne nous intéressons pas à l'intérieur des artéfacts de modélisation et donc ne considérons pas cette relation d'instantiation.

Nous pensons que l'identification des usages de ces deux niveaux de modélisation peut servir de base à la définition d'exigences pour les ateliers de modélisation. Nous présentons donc dans la partie 2 des situations rencontrées lors de travaux de modélisation en les illustrant avec des exemples concrets. Nous abordons rapidement la composition de ces situations (3). Puis nous esquissons une comparaison (5) de différents outils comme des applications de dessin, des ateliers de modélisation ou de méta-modélisation avec l'atelier Openflexo (4) qui met en œuvre l'approche de fédération de modèles. Nous comparons notre approche avec d'autres travaux dans la partie 6 avant de conclure.

2. Situations de modélisation

Pour décrire les situations de modélisation, nous avons deux types d'artéfacts à considérer : les modèles et les méta-modèles. Les situations varient selon l'ordre dans lequel ces artéfacts apparaissent ou sont reliés dans la démarche. Nous simplifions la description en ne considérant qu'un seul acteur dans chaque situation. Une analyse plus fine de ces situations pourraient être intéressante en prenant en compte différents acteurs et donc différentes intentions dans le processus. Les figures 1 et 2 présentent les 11 situations considérées. Chacune des situations est décrite en détail et illustrée dans une des sous sections suivantes par un exemple concret issu de la géométrie.

1) Un méta-modèle existe, on cherche produire un modèle [*concrétisation*].

2) Un modèle existe, le travail consiste à trouver un méta-modèle [*abstraction*].

3) Un modèle existe, il faut trouver plusieurs méta-modèles [*multi-abstraction*].

4) Des modèles existent, il faut élaborer un méta-modèle [*co-abstraction*].

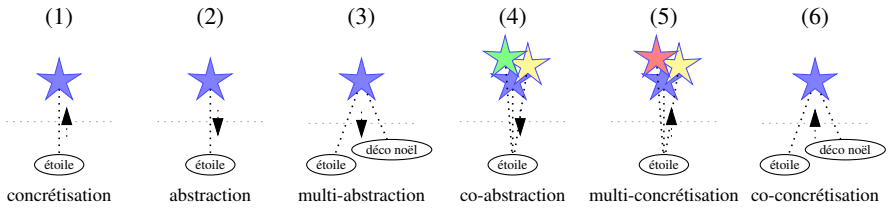


Figure 1. *Des situations de modélisation (haut : niveau instance, et bas : niveau méta)*

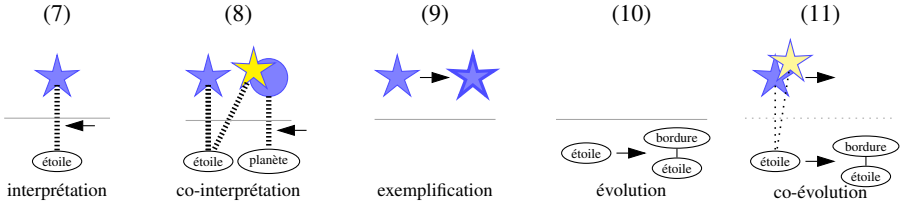


Figure 2. *Des situations de modélisation (haut : niveau instance, et bas : niveau méta)*

5) Un méta-modèle existe, le travail consiste à construire plusieurs modèles [*multi-concrétisation*].

6) Des méta-modèles existent, le travail consiste à construire un modèle [*co-concrétisation*].

7) Un modèle et un méta-modèle existent, il faut les relier [*interprétation*].

8) Des modèles existent, des méta-modèles existent, le travail consiste à les relier [*co-interprétation*].

9) Un modèle existe, le travail consiste à construire un autre modèle (sans aucun méta-modèle) [*exemplification/extension*].

10) Un méta-modèle existe, le travail consiste à construire un autre méta-modèle (sans aucun modèle) [*évolution/extension*].

11) Un méta-modèle existe avec plusieurs de ses modèles conformes, le travail consiste à faire évoluer le méta-modèle (cas précédent) en adaptant (ou non) ses modèles [*co-évolution*].

Les cas (9) et (10) sont probablement equivalents dans le cadre d'une interprétation multi-niveaux de la modélisation. En effet, à un niveau donné d'abstraction, l'absence de référence à un autre niveau, plus concret ou plus abstrait, rend le travail équivalent. Nous les différencions tout de même car la plupart des outils proposent des moyens de manipulations de ces deux niveaux très différents, liés à leur mode de représentation.

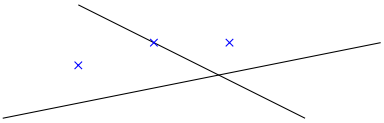


Figure 3. *Des éléments géométriques (points et droites)*

2.1. *Concrétisation*

La concrétisation est la forme d'utilisation la plus classique d'un outil de modélisation. Les concepteurs de l'outil ont préparé des éditeurs adaptés à un ensemble de concepts (méta-modèle). Les utilisateurs élaborent des modèles en respectant les règles prévues. Cette approche est efficace lorsque le modèle que l'on souhaite construire est adapté au méta-modèle utilisé, i.e. les concepts nécessaires sont présents. Dans le cas où les concepts ne s'alignent pas exactement, les utilisateurs sont parfois amenés à tordre les interprétations ou imaginer des usages des concepts non prévus.

Exemple 1 *La figure 3 illustre un modèle d'éléments géométriques, instances des concepts* `point` *et* `droite`*.*

2.2. *Abstraction*

Ce cas, très classique, également dans une démarche de modélisation consiste à partir d'un exemple à identifier les concepts et les règles pour construire un méta-modèle auquel le modèle initial est conforme. Le résultat est exploité par des développeurs d'outil pour construire des éditeurs de modèles.

Exemple 2 *À partir de la figure 3, on peut proposer un méta-modèle* `géométrie`*. Pour illustrer et faciliter la compréhension, nous faisons apparaitre les concepts (éléments de modèles) de* `point` *et de* `droite`*. Le méta-modèle pourrait définir une droite comme un ensemble de points, que tous les points situés entre 2 points quelconques d'une droite appartiennent à cette droite. On notera que « situé entre » n'est pas défini, et que l'espace considéré est implicitement un plan.*

2.3. *Multi-abstraction*

Ce cas, moins classique, généralise la situation d'abstraction, où le but est, à la fois d'abstraire, mais également d'organiser l'abstraction en faisant apparaitre des points de vue complémentaires sur le modèle (2 dans la situation 3 de la figure 1). Nous ne différencions pas les méta-modèles, l'un peut pré-exister ou ils peuvent être conçus tous en même temps. L'important est qu'il existe plusieurs méta-modèles.

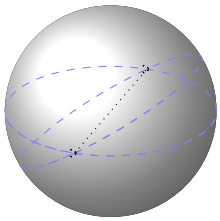


Figure 4. *D'autres points et droites*

Exemple 3 *À partir de la figure 3, il est possible de construire plusieurs méta-modèles. En complément de celui imaginé dans l'exemple précédent, nous pourrions utiliser les concepts de* `bâtons` *(les traits) et* `cailloux` *(les « points ») dans le but, par exemple, de modéliser un jeu.*

2.4. *Co-abstraction*

Cette pratique est une généralisation de la deuxième. Plusieurs modèles servent d'exemples pour construire un méta-modèle auxquels tous les exemples seront conformes. Les démarches taxinomiques ou de classification sont des situations de co-abstraction. L'intérêt d'utiliser plusieurs modèles exemples permet de confronter les concepts et de faire apparaitre des consensus ou, au contraire, des différences d'interprétation et des conflits conceptuels.

Les situations de co-abstraction et de multi-abstraction sont combinables en une multi-co-abstraction non représentée. Nous discuterons de « composition » des situations à la fin de cette partie (3).

Exemple 4 *À partir des figures 3 et 4, il est possible de construire un méta-modèle plus général de la notion de point et de droite. En effet, sur une sphère, on peut interpréter un grand cercle par le concept de* `droite` *et l'intersection de deux grands cercles par le concept de* `point` *(qui en serait deux dans l'interprétation classique).*

2.5. *Multi-concrétisation*

Ce cas est une généralisation du premier. Le même méta-modèle est utilisé à plusieurs reprises pour produire différents modèles. Le risque est que lorsque les interprétations ne sont pas les mêmes pour produire les modèles, des incohérences peuvent apparaitre sans être détectables. Le risque de cette approche est d'autant plus important que les interprétations du méta-modèle sont nombreuses comme c'est le cas avec UML et ses multiples points de variation sémantique (OMG, 2007). Par exemple, la sémantique des diagrammes d'états n'est pas définie précisément en UML et peut donner lieu à diverses interprétations (Chauvel et Jézéquel, 2005).

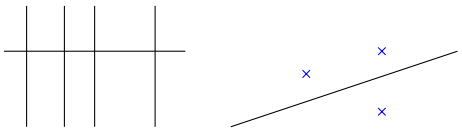


Figure 5. *Des modèles produits à partir des concepts point et droite*

Exemple 5 *À partir des concepts* `point` *et* `droite`*, on peut produire les modèles de la figure 5.*

2.6. *Co-concrétisation*

Cette situation se rencontre lorsque plusieurs experts, chacun disposant de son propre méta-modèle, se réunissent pour décrire un système dans lequel leurs points de vue doivent être représentés. Dans cette situation on *souhaite* laisser les abstractions séparées et ne pas construire un méta-modèle commun.

Exemple 6 *On peut imaginer, pour représenter un bovin, opter soit pour le point de vue d'un boucher, soit pour le point de vue d'un vétérinaire. Bien que pouvant paraître proche (partage de certains découpages anatomiques), les deux spécialités sont cependant suffisamment éloignées pour que la description du même animal ne puissent pas être partagée (fonctions et propriétés des organes différentes).*

2.7. *Interprétation*

Ce cas, s'il est moins habituel, participe sans aucun doute à la démarche de modélisation, au moins dans la tête des « modélisateurs ». Un modèle exemple existe ainsi qu'un méta-modèle. Cette situation est courante et s'appuie sur la connaissance des concepts et de leur organisation par un expert. Le travail consiste à relier les éléments du modèle exemple à un concept présent dans le méta-modèle dans le but :

– soit de s'assurer que l'exemple respecte les règles du méta-modèle ;

– soit de trouver un contre-exemple (le modèle exemple) au méta-modèle afin de le faire évoluer.

Exemple 7 *La figure 6 montre une interprétation d'un dessin existant, pour un méta-modèle (représenté ici dans une version simplifiée par deux boites*[2]*).*

2. On notera qu'un méta-modèle, peut être un objet d'intérêt et devenir lui-même un modèle s'appuyant sur une ou des représentations. La circularité de ces concepts ne simplifie ni leur description, ni leur usage.

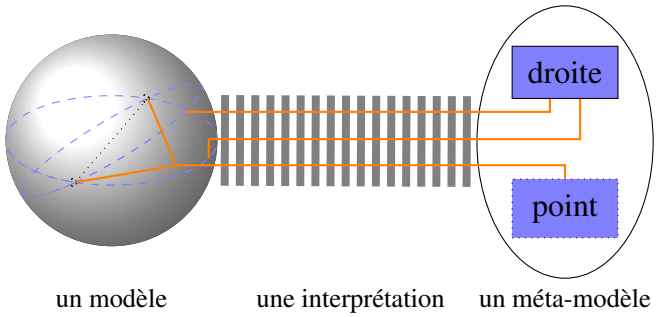


Figure 6. *Une interprétation (l'ensemble des traits pointillés) : deux droites et leur point (unique) d'intersection*

L'exemple 3 (la multi-abstraction) montre l'importance de l'interprétation. Un grand cercle peut être interprété comme une droite. Mais alors, une droite « classique » n'a plus de sens. Un éditeur de dessin pourra dessiner l'une ou l'autre des interprétations, mais si les deux sont mélangées, il y a des risques d'erreur d'interprétation.

Il faut noter que la relation d'interprétation n'est pas la relation d'instantiation. L'instantiation repose sur un mécanisme de construction d'une représentation particulière. Un même concept peut être instancié de multiples manières. L'interprétation décrit une intention, celle de mettre en relation une représentation et son sens, défini et décrit dans un méta-modèle.

2.8. *Co-interprétation*

Ce cas est une généralisation du cas précédent. Plusieurs modèles exemples existent ainsi que plusieurs méta-modèles. Cette situation est encore plus courante que la précédente et s'appuie sur la connaissance des concepts et de leurs organisations par un expert. Le travail consiste à relier les éléments des modèles à un (ou plusieurs) concept(s) présent(s) dans les méta-modèles dans le but :

– soit de s'assurer que les exemples respectent les règles des méta-modèles ;

– soit de trouver un contre-exemple aux méta-modèles afin de les faire évoluer.

Exemple 8 *Les interprétations de* `droite` *dans les géométries proposées en exemple ne sont pas compatibles, alors que celles de* `droite` *et* `baton` *le sont probablement.*

2.9. *Exemplification/Extension*

Cette situation est souvent la première rencontrée. Il s'agit de produire des exemples de représentation qui servent de base à la réflexion. Pour des dessins ou des dia-

grammes, on choisira des couleurs, des formes, des positions relatives des formes pour représenter le problème. L'excellent article de D. Moody (Moody, 2009) passe en revue de nombreux exemples de représentations graphiques. Une histoire des représentation est présentée par M. Friendly dans (Friendly, 2005 ; Friendly et Denis, 2001). Pour les langages, une variabilité extraordinaire des langues et grammaires est observable dans le dictionnaire des langues (Peyraube *et al.*, 2010).

Exemple 9 *Sur la partie gauche de la figure 5, on pourrait introduire une représentation d'un angle droit pour signifier que la droite coupe perpendiculairement une autre droite. La signification « perpendiculaire » est une intention, non encore identifiée dans un méta-modèle, mais qui vient enrichir la notation.*

2.10. *Évolution/Extension*

Cette situation, parallèle à la précédente, ne peut être mise en place qu'une fois la conceptualisation réalisée, c'est-à-dire lorsqu'un méta-modèle est disponible. Il s'agit de faire évoluer les concepts ou les règles les gouvernant. Ce travail se réalise souvent - explicitement ou non - en référence à des évolutions des modèles.

Exemple 10 *Parallèlement au cas précédent, on pourrait enrichir le méta-modèle* `point/droite` *avec un concept d'*`angle` *(ou d'angle droit).*

2.11. *Co-évolution*

C'est une situation liée à la précédente, où le méta-modèle évolue et était relié à des modèles existants ; qu'advient-il des modèles existants ? L'article (Sprinkle et Karsai, 2004) est l'un des premiers à identifier le problème avec le vocabulaire de l'ingénierie dirigée par les modèles.

Ce cas est très complexe car il peut prendre de nombreuses formes. Par exemple, si un nouveau concept est introduit dans le méta-modèle sans qu'aucune instance de ce concept n'ait été précédemment créée, la solution est triviale. Par contre, si la structure d'un concept est remise en cause et que des instance avaient été créées, la solution pourrait ne pas avoir de solution automatisable et demander l'intervention d'un humain pour guider l'évolution.

Nous identifions cette situation, mais ne l'illustrons pas et la laissons à de futures investigations.

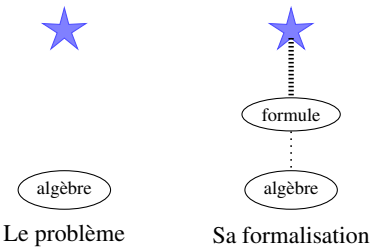


Figure 7. *Une composition pour représenter une approche algébrique.*

3. Composition

Une fois des situations de modélisation identifiées, la question de leur composition se pose naturellement. Nous ne ferons qu'illustrer quelques exemples relevant de ce large problème :

– *Composition comme séquence de situations* : la situation de multi-concrétisation (respectivement multi-abstraction) peut être vue comme une composition de plusieurs concrétisation (resp. abstraction). Ce n'est pas nécessairement le cas, car le fait de regrouper plusieurs actions dans la même situation ne représente pas exactement la même situation que de reproduire *indépendamment* plusieurs fois l'action.

– *Composition comme empilement* : un modèle peut parfois être interprété comme un méta-modèle et conduire à la création de modèles de « niveaux » différents. Inversement, un méta-modèle peut aussi être considéré comme la concrétisation d'un méta-méta-modèle. Il est donc possible d'empiler les situations de concrétisation, abstraction ou interprétation. La figure 7 montre une formulation de la démarche d'algébrisation en 2 étapes. La première (à gauche) consiste à identifier un problème (le modèle) et une théorie (le méta-méta-modèle). La seconde étape (partie droite de la figure 7) consiste à produire une formule qui représente le problème qui est une concrétisation de la théorie et une interprétation du modèle pour le problème considéré. On voit ici qu'on combine (et empile) deux situations élémentaires : concrétisation et interprétation.

– *Composition de modèles* : l'approche que nous avons choisie ne prend pas en compte la structure des modèles et des méta-modèles. Pourtant pour illustrer l'interprétation (figure 6), nous avions fait apparaitre le contenu d'un modèle de géométrie avec `point` et `droite`. Ces *éléments de modèle* peuvent, bien entendu, eux-mêmes être considérés comme des modèles. Ceci peut également être appliqué aux méta-modèles ; des éléments de méta-modèle sont des méta-modèles. Cette relation de composition peut amener à de nouvelles situations que nous ne considérons pas ici.

Notons que ces trois formes de composition se composent. Cet article ne présente qu'une ébauche de l'étude des situations de modélisation et de leur composition qu'il conviendra d'approfondir.

4. Openflexo – L'éditeur « Free Modeling »

4.1. *L'infrastructure de modélisation Openflexo*

Le projet Openflexo vise à construire une infrastructure logicielle dédiée à la construction d'ateliers adaptables. Son architecture modulaire est complétée par une méthode d'assemblage dirigée par les modèles, qui offre une grande souplesse pour la construction d'environnements de modélisation sur mesure. Cette approche, nommée *Diatomée*, s'appuie sur les capacités des composants de l'infrastructure :

– un langage pour la connection dynamique de graphes d'objets (*Connie*) ;

– un *framework* de modélisation supportant l'héritage multiple en Java (*Pamela*) ;

– une API pour la construction d'outils de représentation graphique (*Diana*) ;

– un *framework* de construction et l'interprétation de modèles d'IHM (*Gina*).

Tous ces éléments sont fortement configurables, ce qui permet de construire simplement des ateliers souples et dynamiques. Leur développement est dirigé par la volonté de fournir aux utilisateurs des outils offrant plus de liberté dans la modélisation de systèmes complexes hétérogènes. Tous les composants sont livrés sous license *Opensource* et sont écrits en Java.

4.2. *La fédération de modèles*

Le cœur de l'infrastructure (*openflexo core*) contient des composants dédiés à la mise en oeuvre de la *fédération de modèles*. Cette technique autorise la construction de nouveaux modèles (représentations graphiques, documents, ...) à partir d'un ensemble hétérogène de sources de données[3] (vues comme des modèles). Elle peut être mise en œuvre pour outiller aussi bien les approches de type *Architecture Framework* (Schekkerman, 2004), que la modélisation multi-paradigmes (Amaral *et al.*, 2010).

Dans sa version actuelle (*Semantics+ 1.6.1*), l'atelier supporte la création conjointe de modèles conceptuels fédérés (syntaxe abstraite) et des représentations graphiques associées (syntaxe concrète). Les diagrammes construits par l'utilisateur peuvent mélanger dessins libres et formes pré-définies. Tous les éléments du niveau méta (modèle conceptuel et formes graphiques prescrites) peuvent être modifiés en cours d'exécution. Cela permet de faire émerger de nouveaux concepts à partir des formes libres (non-associées à un concept) présentes sur les vues. Le détail de l'approche est décrit dans (Guychard *et al.*, 2013).

3. Actuellement possible avec des ontologies (OWL), des modèles EMOF/EMF, des documents XML et des tables Excel.

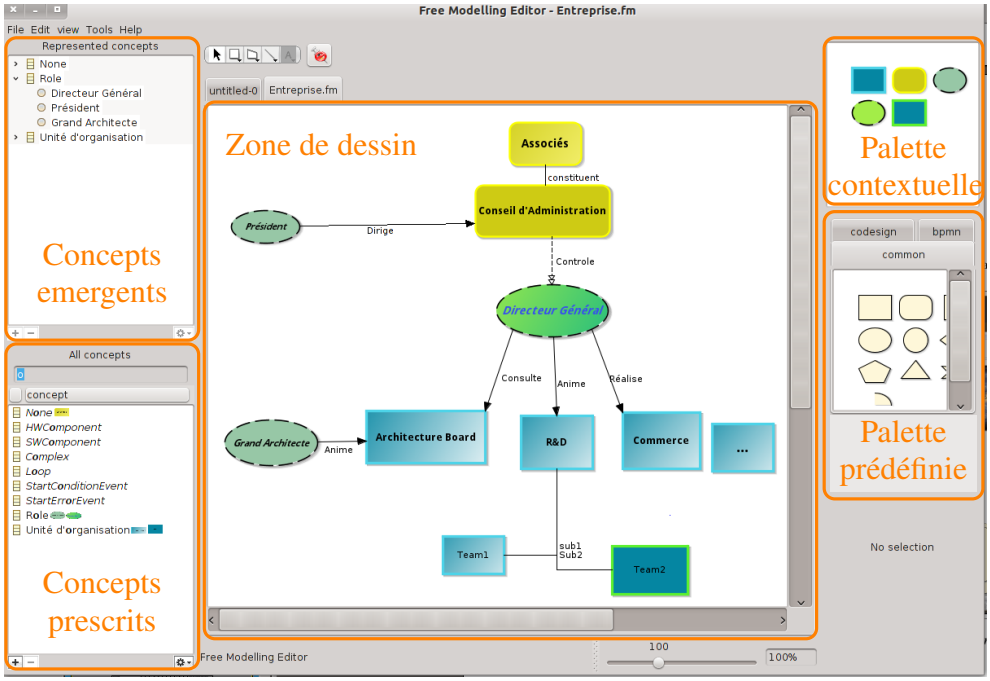


Figure 8. *L'organisation du «Free Modeling Editor»*

4.3. *L'outil de « modélisation libre »*

Assemblé selon les principes de *Diatomee* décrits ci-dessus, le *« Free Modeling Editor »* (FME) est un prototype destiné à l'expérimentation de nouveaux modes d'intéraction avec les modèles. Il est utilisé pour faciliter l'émergence d'une syntaxe graphique simultanément au modèle conceptuel associé. Son interface, illustrée par la figure 8, est découpée en une partie *outils de dessin* sur la droite et une partie *modèle conceptuel* sur la gauche. L'utilisateur peut librement dessiner (situation d'*exemplification*) dans l'espace au centre en plaçant sur la page de travail des formes graphiques libres sélectionnées depuis une des palettes « métier » ou en ré-utilisant des formes définies dans ce contexte (palette « contextuelle »). Lorsqu'une forme est sélectionnée, il est possible (via un menu contextuel) de l'associer à un élément du modèle conceptuel présenté sur la gauche (*interprétation*). Si le concept correspondant n'existe pas encore, l'utilisateur peut en définir un nouveau (*abstraction*).

Il est facile d'expérimenter les différentes situations de modélisation décrites ci-avant avec le *FME*. La séparation des espaces de travail et des responsabilités conduit à un fonctionnement de l'atelier qui permet de pratiquer les 10 premières situations présentées. Toutefois, le modèle conceptuel utilisé est simpliste et on ne peut pas établir de relations structurelles ou sémantiques entre concepts.

L'outil est actuellement autonome et indépendant de l'infrastructure de fédération de modèles. Un travail est en cours pour les faire coopérer. Cela facilitera la création « à la volée » de vues définies par l'utilisateur au dessus de modèles multi-niveaux, complexes et distribués. Nous pensons que ce fonctionnement conjoint nous permettra d'outiller l'ensemble des 11 situations présentées dans cet article.

5. Évaluation d'outils

Nous considérons comme outil de modélisation des outils de dessin ou de présentation comme dans `Libreoffice`, des outils de modélisation avec méta-modèle comme des éditeurs UML, des ateliers de méta-modélisation comme MetaEdit+, et notre outil de fédération de modèle `Openflexo`.

Les situations présentées précédemment servent de critère de comparaison pour chacun des types d'outils. Il s'agit d'une esquisse de comparaison ; des ateliers ou des outils pourraient offrir des capacités spécifiques non étudiées dans cet article.

Outils	1	2	3	4	5	6	7	8	9	10	11	Commentaires
Dessin (Libreoffice, etc)	–	–	–	–	–	–	–	–	+	+	–	sans méta-modèle
Dédié (UML, BPMN, etc)	+	(1)	(1)	(1)	+	–	–	–	+	(1)	–	avec méta-modèle
Meta-Editeur (MetaEdit+, etc)	+	+	(2)	+	+	(2)	(2)	(2)	+	+	+/-	interprétation stricte
Openflexo	+	+	+	+	+	+	+	+	+	+	?	fédération

Commentaires :

(1) Pour UML, les profils s'en rapprochent peut-être…

(2) On peut probablement le faire en programmant…

Les outils de dessins ne permettent pas de travail de méta-modélisation ; ils restent au niveau de la syntaxe concrète. Les outils dédiés sont spécialisés pour traiter les modèles prévus, mais n'offrent pas d'outils pour changer ou élargir le point de vue. Les méta-éditeurs permettent d'enrichir les points de vue mais dans une approche méta vers instance et sans offrir d'outils de composition. Openflexo propose, à partir d'une analyse des besoins des situations de modélisation, un atelier plus complet de manipulation de modèles.

6. Travaux connexes

Dans (El Kouhen *et al.*, 2012), les ateliers *Generic Modeling Environment* (GME), *Rational Software Architect*, MetaEdit+, *Obeo Designer* et Eclipse GMF sont comparés selon 8 critères : niveau de personnalisation, expressivité graphique, complétude graphique, ouverture, utilisabilité, personnel requis, type de license, caractéristique des artéfacts. La synthèse sur l'utilisabilité qui est au centre de notre article est :

> « The best usability is offered by far by the Obeo Designer editor in terms of efficiency, accessibility, satisfaction and overall number of features. (. . .)The manipulation of elements is somewhat awkward in RSA and GME. »

Les comparaisons ne font pas référence à des situations de modélisation comme nous proposons de le faire, mais à des capacités ergonomiques. Les situations implicitement rencontrées sont la concrétisation, l'abstraction et la co-abstraction, et la multi-concrétisation.

Dans (Amyot *et al.*, 2006), les ateliers *Generic Modeling Environment* (GME), *Telelogic Tau* G2, *Rational Software Architect* (RSA), XMF-Mosaic, Eclipse EMF+GEF sont comparés selon 6 critères : complétude graphique, utilisabilité des éditeurs, efforts requis, évolution des langages, intégration et outils d'analyse et de transformation. Implicitement, les cas de concrétisation sont évoqués au travers de l'ergonomie, et la seule situation de modélisation explicitement évoquée est la co-évolution.

Dans (Kirchner et Jung, 2007), les méta-outils MetaEdit+ et Cubetto sont comparés à l'aide d'une analyse multi-critères très simple prenant en compte : coût, désinstallation, capacité de modélisation avec des langages de modélisation prédéfinis, l'utilisation d'approches comme *Event-driven process chains* (EPC), *Petri nets* (PN), *Unified Modeling Language* (UML) et *Entity-Relationship-Model* (ERM), la capacité de simulation et l'usage de métriques. Il n'y a pas de référence explicite à des situations de modélisation autre que les cas de concrétisation et multi-concrétisation.

Un article de synthèse sur la méta-modélisation dans la conception et l'optimisation est réalisé dans (Wang et Shan, 2007). De nombreuses situations de modélisation sont identifiés : l'approximation de modèle, l'exploration de l'espace de conception, la formulation de problèmes, et la résolution de différents types de problèmes d'optimisation (que nous ne considérons pas car elles peuvent être outillées indépendamment). Mais la définition de la méta-modélisation assez déroutante : « The simple model is often called metamodel. »

À aucun moment, la grammaire ou le méta-modèle, au sens où nous l'entendons, n'est intégré dans le processus de modélisation ou de méta-modélisation. Pourtant une table faisant référence à des méta-modèles communément utilisés cite (entre autres) : *Polynomial* (*linear*, *quadratic*, or *higher*), *Splines* (*linear*, *cubic*, NURBS), *Knowledge Base or Decision Tree* qui sont bien des méta-modèles au sens où nous l'entendons. Les exemples de l'article décrivent des situation de type concrétisation et multi-concrétisation, interprétation et co-interprétation, et exemplification.

Enfin, dans (Muller *et al.*, 2010) les auteurs étudient d'autres relations entre modèles en formalisants différentes intentions dans le but de « faciliter le raisonnement et la discussion sur les méta-modèles et leurs relations » et de servir de base au développement d'ateliers de méta-modélisation « conscients » des intentions.

7. Conclusion

La modélisation est une démarche complexe, avant tout intellectuelle, mais qui peut être outillée. Le papier et le crayon (ou leur équivalent) restent des outils de base grace à leur flexibilité et à la liberté qu'ils offrent. L'outillage informatique apporte des fonctionnalités supplémentaires qui permettent de vérifier des propriétés telles que la conformité. Pourtant, cet apport se fait au détriment de la liberté. Comment concilier vérification et liberté ? L'identification de situations de référence de modélisation peut servir de spécification d'usage pour des ateliers de modélisation. Nous avons observé :

– qu'il ne faut pas réduire la modélisation au dessin ; la notion de méta-modèle apporte des outils de vérification et peut guider lors de la concrétisation ;

– qu'un méta-modèle se construit et doit pouvoir évoluer ; on doit pouvoir aller au-delà de ce qui est prévu ;

– que méta-modèle et modèle ne sont pas nécessairement liés a priori, mais que c'est une activité de modélisation que d'identifier ces liens.

Modéliser requiert toutes les situations, combinées dans des ordres variés. Par exemple, on peut commencer par exemplifier, puis co-abstraire et interpréter, puis généraliser, pour exemplifier à nouveau afin de valider le méta-modèle. Un atelier de modélisation doit faciliter le travail à tous ces niveaux : intra-niveau et inter-niveaux.

La plupart des outils actuels sont trop contraignants – ils obligent à adopter un méta-modèle (ou un ensemble de méta-modèles) figé – ou trop complexes – et le travail sur le méta-modèle est une activité de programmation qui ramène souvent les concepts aux paradigmes des informaticiens.

L'infrastructure Openflexo propose une approche multi-niveaux par assemblage ou fédération de modèles. Il peut être utilisé comme simple outil de dessin, comme un outil de modélisation classique, mais aussi comme un outil de méta-modélisation pour élaborer un méta-modèle sans programmation et enfin pour mettre en relation – interpréter – des dessins et des méta-modèles.

Pour prolonger ce travail, nous envisageons une analyse plus fine des situations, en prenant en compte par exemple les différents acteurs, mais aussi des situations non pas constructives, mais destructives comme la suppression d'une interprétation ou d'un exemple. D'autres situations sont certainement intéressantes comme le changement de niveau évoqué dans la partie 3, quand un modèle devient un méta-modèle ou réciproquement, ou la composition de modèle évoquée dans cette même partie. Enfin, une étude précise sur des ateliers ou des outils de modélisation devrait être entamée.

8. Bibliographie

Amaral V., Hardebolle C., Karsai G., Lengyel L., Levendovszky T., « Recent Advances in Multi-paradigm Modeling », *Models in Software Engineering*, vol. 6002 de *Lecture Notes in Computer Science*, p. 220-224, Springer, 2010.

Amyot D., Farah H., Roy J.-F., « Evaluation of Development Tools for Domain-Specific Modeling Languages », *System Analysis and Modeling : Language Profiles*, vol. LNCS 4320, p. 183–197, Springer, Berlin, Heidelberg, 2006.

Chauvel F., Jézéquel J.-M., « Code Generation from UML Models with Semantic Variation Points », Briand L., Williams C., Eds., *Model Driven Engineering Languages and Systems*, vol. 3713 de *Lecture Notes in Computer Science*, p. 54-68, Springer, 2005.

El Kouhen A., Dumoulin C., Gerard S., Boulet P., « Evaluation of Modeling Tools Adaptation », rapport, juin 2012, Laboratoire d'Informatique Fondamentale de Lille (LIFL).

Friendly M., Denis D. J., « Milestones in the history of thematic cartography, statistical graphics, and data visualization », Web document, `http://www.datavis.ca/milestones/`, 2001.

Friendly M., « Milestones in the History of Data Visualization : A Case Study in Statistical Historiography », Weihs C., Gaul W., Eds., *Classification : The Ubiquitous Challenge*, p. 34–52, Springer, New York, 2005.

Guychard C., Guerin S., Koudri A., Dagnat F., Beugnard A., « Conceptual interoperability through Models Federation », *Semantic Information Federation Community Workshop, Models Conference*, 2013.

Kirchner L., Jung J., « A Framework for the Evaluation of Meta-Modelling Tools », *The Electronic Journal Information Systems Evaluation*, vol. 10, n° 1, 2007, p. 65–72.

Kleppe A., « A Language Description is More than a Metamodel », *Fourth International Workshop on Software Language Engineering*, Grenoble, France, October 2007, megaplanet.org.

Lachapelle C., Cunningham C., « Engineering Is Elementary : Children's Changing Understandings of Engineering and Science », *American Society for Engineering Education Annual Conference & Exposition*, Honolulu, HI, June 2007.

Moody D., « The "physics" of notations : toward a scientific basis for constructing visual notations in software engineering », *IEEE Transactions on Software Engineering*, vol. 35, n° 6, 2009, p. 756–779.

Morris W. T., « On the art of modeling », *Management Science*, vol. 13, n° 12, 1967, p. B707–B717.

Muller P.-A., Fondement F., Baudry B., « Modeling modeling modeling », *Software and Systems Modeling*, , 2010.

« UML 2.0 Superstructure Specification », `http://www.omg.org/cgi-bin/doc?ptc/2003-08-02`, 2007.

Peyraube A., Bonvini E., Busuttil J., *Dictionnaire des langues*, Presses Universitaires de France, 2010.

Schekkerman J., « A Comparative Survey Of Enterprise Architecture Frameworks », rapport, 2004, Institute For Enterprise Architecture Development/Capgemini.

Sprinkle J., Karsai G., « A domain-specific visual language for domain model evolution », *Journal of Visual Languages and Computing*, vol. 15, n° 3, 2004, p. 291–307.

Wang G. G., Shan S., « Review of metamodeling techniques in support of engineering design optimization », *Journal of Mechanical Design*, vol. 129, 2007, page 370.

Experimentation of a Graphical Concrete Syntax Generator for Domain Specific Modeling Languages

Blazo Nastov[1], François Pfister[1]

1. LGI2P, Ecole des Mines d'Alès, Parc Scientifique G. Besse, 30000 Nîmes, France
{Blazo.Nastov, Francois.Pfister}@mines-ales.fr

ABSTRACT. *Graphical Domain Specific Modeling Languages (DSML) are alternatives to general purpose modeling languages e.g. UML or SysML. They describe models with concepts and relations specific to a domain. Defining such languages consists of defining an abstract syntax and a graphical concrete syntax accompanied by a correspondence mapping between the elements of each one. Such process is composed of two phases: the abstract syntax definition and the concrete syntax definition. This paper describes concepts and mechanisms allowing to guide and to assist an expert from any engineering domain to define and formalize the concrete syntax of a graphical DSML considered as relevant in this domain. We define multiple classifications of the abstract syntax elements based both on the abstract syntax and on the concrete syntax. Grounded on those classifications, we present how a part of the concrete syntax can be generated automatically from an abstract syntax by a graphical role election.*

KEYWORDS: *MDE, DSML, model, metamodel, languages engineering*

1. Introduction

Complex system engineering is an approach for designing complex systems based on creating, manipulating and analyzing various models. Each model is related to and is specific to a domain (e.g. quality model, requirements model or architecture model). Classically, models are the subject of study of Model Driven Engineering (MDE) (Kent 2002) and they are nowadays built by using, and conforming to Domain Specific Modeling Languages (DSMLs). Creating DSML consists in defining its abstract and concrete syntaxes. An abstract syntax is represented by a metamodel composed of classes representing modeling concepts and relationships between classes representing relations and dependencies between modeling concepts. The literature highlights two ways for defining an abstract syntax. Either extending UML (OMG 2011) (UML profile (Fuentes-Fernández & Vallecillo-Moreno 2004)) or deriving a metamodel directly from MOF (MOF 2002). Various *concrete syntaxes* define the representations of modeling concepts which are either graphical or textual.

This paper describes an experimentation of concepts and mechanisms allowing to guide and to assist an expert from any engineering domain to define and formalize a graphical concrete syntax for a given DSML considered as relevant in this domain.

A graphical concrete syntax is composed of graphical elements, each one representing how a given abstract syntax element (class or relationship) is graphically rendered for the end user. Basically, classes are represented as diagram nodes with one exception, representing a class as diagram edge detailed furthermore in the paper, and relationships are represented as diagram edges. Such representations can be refined by size, shape, role, color, etc. Once defined, they are mapped to abstract syntax elements. The mapping is called *correspondence mapping*. Together, the abstract syntax, the concrete syntax and the correspondence mapping, form a DSML syntax (Kleppe 2007). Such syntax allows creating models seen as graphical representations of a part of a modeled system.

The correspondence mapping is usually source of errors and abstract syntax elements may have inconsistent graphical representations, for instance visual information representing "node" mapped to relationship. This is either interpreted as a mapping error, or the considered relationship is excluded from the generated editor palette. We aim to reduce such errors based on possible graphical representation of modeling concepts which we refer to as "graphical roles". We classify abstract syntax elements considering low levels of graphical representation, distinguishing if a given element would be graphically represented as node or edge or if it would not be graphically represented at all. High level graphical representation such as size, shape, color, etc. is out of reach of this paper and will not be discussed. Abstract syntax elements have one or multiple graphical roles. We generate automatically a part of the concrete syntax information by choosing graphical roles. The generated concrete syntax information is one "elected" possibility out of all different possible graphical representations.

This paper is structured as follows. Section 2 presents and discusses some existing works in the domain. Section 3 details the first contribution of this work i.e. the different classifications relevant for characterizing an abstract syntax element. Section 4 details the second expected contribution i.e. a semi-automatized process for concrete syntax generation before concluding about research perspectives.

2. State of the Art

This section discusses first the process of creating DSMLs, and then presents different frameworks, each one able to creating graphical DSML and to generate graphical editors.

In practice, designers are focusing on the DSML metamodel that defines the core concepts of the language. This phase is generally well controlled by practitioners, but is often separated from the design of graphical concrete syntax. DSMLs are better built in an incremental way, so an iterative approach is adopted that allows a validation and a verification process, by many language stake-holders and end-users. Such an iterative and incremental process would be possible only with an adapted

tool support (Cho 2011). The workflow described in Figure 1, shows how to process temporally in parallel and with the same interest, the two aspects of the modeling process.

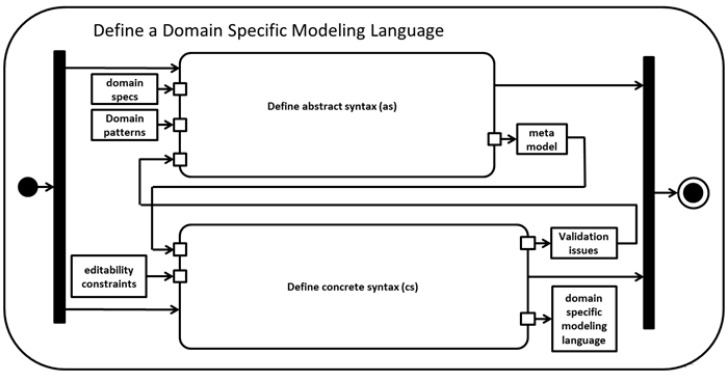


Figure 1. Define a DSML

2.1. Abstract syntax definition

An abstract syntax plays a central role in a language specification. It is the pivot between concrete syntax and semantics. The original meaning from the term abstract syntax comes from natural language, where it means the hidden, underlying, unifying structure of a number of sentences (Chomsky 1965). In computer science, particularly in MDE, an abstract syntax is represented by a metamodel. Meta-modeling languages such as the OMG's standard MOF provide the basic concepts and relationships in terms of which it is possible to describe a metamodel. Nowadays, multiple environments and metamodeling languages help to define an abstract syntax: Eclipse-EMF/Ecore (Steinberg et al. 2008), GME/MetaGME (Ledeczi et al. 2001), AMMA/KM3 (Jouault & Bézivin 2006) or XMF-Mosaic/Xcore (Clark et al. 2008). We use the graphical editor proposed by Eclipse-EMF, describing a metamodel using the EMF's metamodeling language Ecore. An example of a metamodel is shown in the Figure 2, which will be used as core element to our classifications and in a later phase as a source to the semi-automatic concrete syntax generator.

2.2. Concrete syntax definition

Concepts defined by the abstract syntax represent the underlying structure of a language. However, such concepts need additional information about their rendering to the end user. A concrete syntax of a language provides such information. We focus on graphical concrete syntax for DSMLs. The best graphical expressiveness of a graphical language depends on graphical economy of the representation. Moody in

(Moody 2009), presents a criteria for the expressive power of a graphical language. These criteria consider, first, the relative position of graphic symbols that make up the graphical language, and then, stylistic criteria about shapes, icons colors and line styles.

There are indeed several tools for creating a concrete syntax for a given abstract syntax and afterwards generating graphical editors. Graphical Modeling Framework (GMF) (Gronback 2009) is a framework based on a mapping between Ecore and Gef (a Graph drawing engine). This framework is powerful and widely used, but poorly documented. There are a number of frameworks based on top of GMF facilitating the process. TOPCASED propose a tool called "generator for graphical editors" allowing for a given Ecore model to define a graphical concrete syntax and then to generate the proper graphical editor (Pontisso & Chemouil 2006). Eugenia (Kolovos et al. 2010) is a framework based on top of GMF, annotating the metamodel with concrete syntax elements in order to generate the GMF artifacts. Diagraph (Pfister et al. 2013) is a tool for designing graphical DSML respecting the previously described process. The abstract syntax is represented by an Ecore metamodel and the concrete one is derived from the abstract syntax by a transformation targeting a generic metamodel of the graphical concrete syntax. Designing the concrete syntax implies the definition of this transformation; the latter is registered into the abstract syntax trough meta-data (annotations).

2.3. Correspondence mapping

The metamodel, shown on the Figure 2 contains the classes and the relations used to describe respectively the concepts and the relations of the personal computer hardware domain. To get an entire status of a language, this metamodel should be coupled with a textual or a graphical concrete notation. In our case, a concrete syntax should be defined by a set of graphical information e.g. type (node or edge), size, shape, etc. This information should be mapped to the abstract syntax elements. Such mapping is a bijection relationship between an abstract syntax element and a concrete syntax element. It is also called a correspondence mapping. Indeed, it is the correspondence mapping that determines how abstract syntax elements are going to be graphically represented, mapping them to a concrete syntax element (graphical information).

The work presented in this paper is demonstrated and implemented with Diagraph. We propose semi-automatically generated graphical concrete syntax for a given Ecore metamodel as a functionality included in this framework.

3. Classification

Understanding graphical domain specific modeling languages requires some definitions. We propose, in this section, three classifications of abstract syntax elements based on the graphical roles they have in a DSML and a general typology

of graphical elements in DSML. The semi-automatic concrete-syntax generation process will be grounded on our typology.

As an illustration, we start from a toy language that describes a personal computer (PC) (see Figure 2). The whole collection of abstract syntax elements represent, together, the metamodel (the abstract syntax) of the language. Instances of those languages give rise to object graphs (graphs of objects which are instances of the classes contained in the abstract syntax). Such object graphs could be represented in a raw manner, as non-filtered and non-structured basic graphs, devoid of any cognitive power. Graphical modeling languages should provide specific visual representations, enabling their users to have good cognitive perceptions of the expressed statements. Thus, and this is a key concept, it is important to notice that every element of an abstract syntax (class or relationship) will not have a graphical representation. We discuss the ability of abstract syntax elements, to be graphically representable and/or represented. This is a shortcut to tell about the ability, for the instances of those abstract syntax elements, to be representable or represented. Among the elements that are not visible in graphical statements, some of them are not graphically *representable*, due to their role within the abstract syntax, and others are not graphically *represented* according to design decisions, generally for the reason of graphical economy (Moody 2009), or for distributing the language between different conceptual points of view which correspond, each, to a different graphical view.

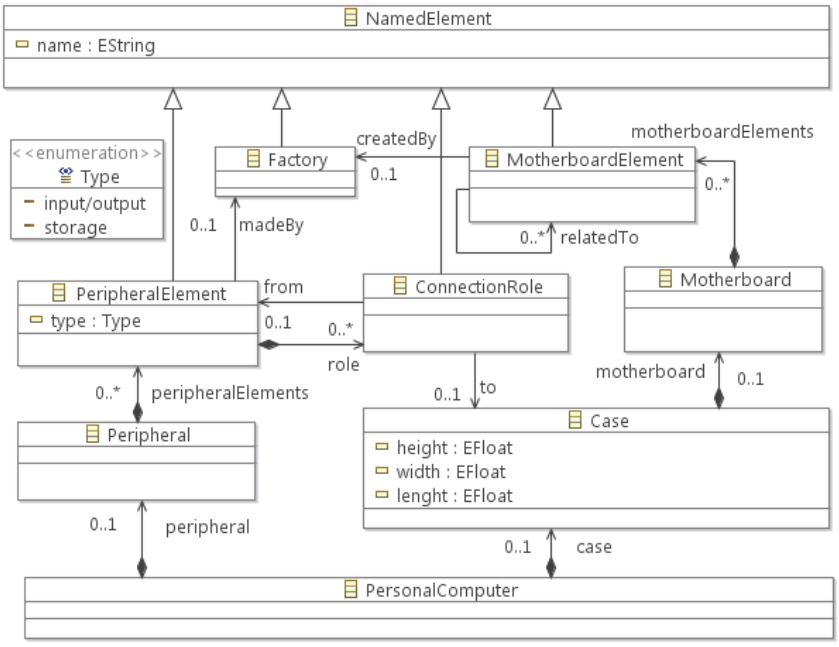


Figure 2. A metamodel describing a personal computer

Note that a graphical language may split the graphical space between several graphical views. Each view should be elected (chosen) by the language designer. As a starting point, we consider only one unique (structural) view for our PC language, which is related to the root class of the metamodel, also known as a *point of view (POV).*

3.1. Graphical representability and Graphical representation

Depending on existing structures in the abstract syntax, the abstract syntax elements (class or relationship) can be classified into *graphically representable elements* and *graphically not representable elements*. Depending on design decisions (presented in a concrete syntax and a correspondence mapping), among the prior representable elements, we will distinguish, for a given graphical view, *graphically represented elements* and *graphically not represented elements*.

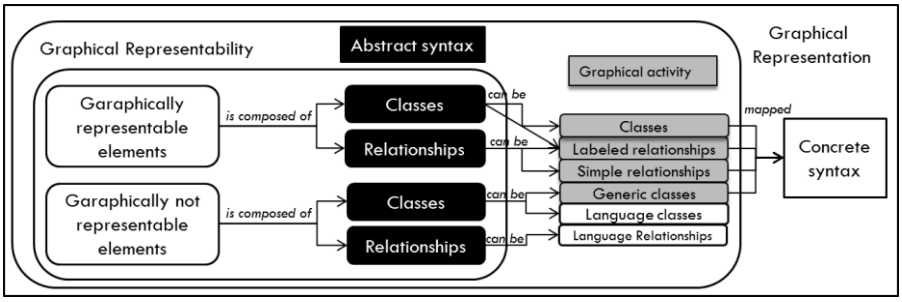


Figure 3. Graphical DSML elements classifications

3.1.1. Graphical representability

Based on the abstract syntax, we can distinguish *graphically representable and not representable classes* and *graphically representable and not representable relationships.* Figure 3 illustrates such classification. The left site of the figure represents graphically representable elements and graphically not representable elements. Each category is composed of graphically representable classes and relationships and graphically not representable classes and relationships. Together, representable elements (classes and relationships) and not representable elements (classes and relationships) form the abstract syntax as shown on the figure by black rectangular forms.

A graphically representable class is an abstract syntax element that is related by a relation of composition to another class, the latter having the role of the container, and if it is *connected* to the root class of the metamodel (the POV). The rest of the entities are identified as graphically not representable due to the inability of being instanced as a part of the language. We say that a class is connected to another class, if a set of navigable classes exists between the last two, e.g. a navigation that begins at the first class, passing through all classes, and finishing at the last. A graphically

representable relationship connects two classes that are representable or subsume a representable class. Here also, the rest of the relationships are identified as graphically not representable. Table 1 represents Figure 2 graphically representable and not representable elements, considering *PersonalComputer* as a root class.

Table 1. Representable and not representable elements.

Representable classes	Not representable classes	Representable relationships	Not representable relationships
Peripheral, PeripheralElement, Case, Motherboard, ConnectionRole, MotherboardElement	Factory, NamedElement	peripheral, case, peripheralElements, role, relatedTo, from, motherboardElements, to, motherboard	createdBy, madeBy

3.1.2. Graphical representation

Based on a correspondence mapping, an abstract syntax and a concrete syntax, abstract syntax elements can be divided into *graphically represented elements* and *graphically not represented elements*. An abstract syntax element is graphically represented if, first, the element is a graphically representable element and, second, if it is mapped by the concrete syntax element (by a design decision). The rest of the abstract syntax elements, called *graphically not represented elements*, will not have a graphical representation, either because of a design decision or because they have a *graphically not representable* status.

Among graphically represented elements we distinguish, *graphically represented classes* and *graphically represented relationships* which are either *simple relationships* or *labeled relationships*. The common characteristic of these elements is that they are all, representable and mapped to the concrete syntax, while the difference lies in their underlying structures. A represented class is a single element inside an abstract syntax. For instance, if a conforming graphical information (concrete syntax element) is mapped to the class *Peripheral* (see Figure 2) the latter become a represented class. A simple relationship is also represented by a single element, a relationship (association or composition) inside an abstract syntax e.g. *motherboard*, and of course a conforming concrete syntax elements that is mapped to the prior. A labeled relationship is a compound structure, represented by a class and its attributes on the abstract syntax side, and by a labeled line on the concrete side, where the labels are mapped to the attributes. Such structure is often referred in the literature as a *class-association* pattern. For instance, the class *ConnectionRole* and the relationships *role*, *from* and *to*, are forming a labeled relationship inside the abstract syntax, between the classes *Peripheral* and *Case*.

Graphically not represented elements, if mapped by a conforming concrete syntax element, might have *indirect impact* of the graphical representation even besides the not representable status. Therefore, among the graphically not

represented elements, we distinguish those that might have indirect graphical impact called *generic classes* (e.g. *NamedElement*) from those not having any possibility of graphical impact called *language classes* (e.g. *Factory*) and *language relationships* (e.g. *createdBy* and *madeBy*).

Generic classes should first, subsume (directly or indirectly) at least one represented class, second, they should hold a property (e.g. attribute) and third it should be mapped to the conforming concrete syntax element. This property is afterwards graphically inherited by the subsumed represented class. Note that the graphical inheritance is represented, in the abstract syntax as an inheritance relationship (direct subsuming) or a set of inheritance relationships (indirect subsuming) between the generic class and the represented class, and in the concrete syntax by a conforming concrete syntax element that is mapped to the property. It is the represented class that represents graphically the inherited graphical property of the generic class and thus the indirect graphical impact of the latter.

Language classes, besides the not representable status, do not subsume any represented class and thus the absence of any graphical impact. Such element has only cognitive and structural purpose in the abstract syntax.

As an illustration, Figure 3 describes a classification of graphical representations and the impact on a graphical concrete syntax.

The classification is illustrated on the left side of the figure while the right side of the figure illustrates a concrete syntax. In the middle (left), the abstract syntax elements are represented by black rectangular forms as representable classes, relationships, not representable classes and relationships. Each of these elements (representable and not) can be classified by the graphical representation classification based on their mapping on the concrete syntax. This is shown on the middle (right) side of the figure.

In order to factorize a graphical property from a generic class, the latter should be contained in the class by the mean of an attribute and/or a relationship (e.g. the attribute *name* from the generic class *NamedElement*). Since the factorization is of graphical nature, the concrete syntax mapping should be enriched with the proper information. Finally, a graphically represented class should be derived by inheritance from that generic class. Such a graphical inheritance mechanism will propagate, at the concrete syntax level, the attributes and the relationships that are inherited at the abstract syntax level. Therefore, the inheritance relationships relating a represented class to a generic class can be considered as *generic relationships*.

3.2. Graphical element activity classification

In the previous section we discussed a classification of graphical roles, distinguishing classes and relationships that are graphically represented or not represented, depending both on their role within an abstract syntax and on design decisions captured in a concrete syntax. If an abstract syntax element is graphically represented, it means that its graphical role is stored somewhere and associated to

that element. All the graphical roles are stored within a concrete syntax model which acts as a layer of meta-data over the abstract syntax, therefore, a mapping should exists between the concrete and the abstract syntax. As we have seen above, there are also classes within the abstract syntax that are used to factorize graphically represented properties called *generic classes*. The inheritance mechanism that implements the factorization should also be noted into the concrete syntax which is mapped on the abstract syntax, in order to apply that design decision. On the other hand, there are classes that do not play any graphical role. This type of classes is called *language classes* and they do not figure within the concrete syntax layer. This classification aims to distinguish the last from the other elements, classifying them into *graphically active* and *graphically passive* elements. As illustration, in the middle (right) side of Figure 3, graphically represented and not represented elements are colored gray in order to represent the graphically active elements. The rest are considered as graphically passive elements.

A graphically active element is a language element that has associated information in a concrete syntax that causes the production of an element within the graphical notation of a domain specific modeling language. Such element is either a *graphically represented element* or a *graphically not represented generic class*. The graphically representable elements and generic classes of any abstract syntax are forming the graphically active elements.

A graphically passive element is a language element that has no associated information in a concrete syntax. Such element is also called an *abstract language element* (it could be a class or a relationship) and is used to hold a language information that is not graphically represented. The language classes and the not represented relationships of any abstract syntax are forming the graphically passive elements.

3.3. Graphical abstract syntax elements typology

It is very important to know what kind of elements can be presented by a graphical DSML and their different graphical representation possibilities. Therefore, this section defines a general typology for the representation of graphical DSML elements. Only graphically represented elements of the abstract syntax have an actual graphical representation (if they are mapped to a concrete syntax element), and therefore, this typology focuses only on graphically represented elements. Among them, we classified represented classes, and represented relationships, which can be either simple relationships or attributed relationships. When representing such elements within a graphical editor we refer to them as a *node, a simple link* and *a labeled link.* For a given (structural) view, we distinguish three different types of nodes, two different types of simple links and labeled links.

We classify the nodes into: *canvas*, *top level nodes*, and *child nodes*. The *canvas* is represented by the root class in the abstract syntax also called *the point of view (POV)*, containing the other classes and graphically represented as a screen (white board) where we add the rest of the nodes. The second type of nodes are the nodes

that are disposed onto the *canvas* or *top level nodes*, while the rest of the nodes, that are embed into top level nodes or into other child nodes are called *child nodes* representing the third type.

Among the simple links we classify *relating links* and *embedding links*. A relating link is graphically represented as a *line* or an *arrow*, relating two nodes. In the abstract syntax it is represented by an association relationship between two classes. An *embedding link* is graphically represented as a *nested mechanism* between two nodes where one of them is container and the other is content. In the abstract syntax it is represented by a composition relationship between two classes (the nesting mechanism can start at the top level node, and the nesting depth is not limited; nodes are *top level nodes* at the first level into the *point of view*, and the deeper nested nodes are *child nodes*). In this case, there is neither line nor arrow, but the structure is, conceptually, an edge of a graph, graphically representing an embedding as a part of the view, the latter being mathematically a graph. This point is specific to our interpretation of diagramming, versus other approaches which consider that diagrams are unstructured drawings composed of graphical elements, related the one with the others but not necessarily as a graph structure, and (as we also do) mapped to a semantic artifact, the metamodel.

A *labeled link* is graphically represented as a *line* or an *arrow*, carrying one ore many labels and relating two nodes. The underlying structure of such label is represented by an attribute of a class and therefore the need of a supplementary class inside the abstract syntax.

4. Building the concrete syntax of a graphical DSML: needs and demonstration

The process of creating graphical DSML requires a language designer that is a double expert. First, an expertise in the domain he intends to model is required. Second, an expertise of creating "well-formed" DSML in the sense of language metamodeling is required.

A domain expert is a person who is an expert in a particular area or topic. This term is frequently used in the area of software development and the term refers to a particular domain (e.g. virtual machines, code generation, etc.) rather than a software domain. For instance, a "virtual machines" expert is a person that has a significant knowledge in the field of developing "virtual machines". The domain expertise is an important prerequisite to the process of creating graphical DSML.

The expertise of creating "well-formed" DSML is on the one hand, in relationship with *metamodeling* and *modeling* (e.g. modeling using UML) and on the other hand, with small amount of concepts representing the *graphical language modeling*. Metamodeling and modeling are not the purpose of this paper and will not be discussed. For more details, see (Omg 2006; Mellor & Balcer 2002; Specification & Bars 2007). Graphical language modeling includes concepts that are specific for this kind of modeling. The expertise of graphical language modeling is knowledge of representing modeling concepts of any DSML i.e. the concepts introduced in

element typology section. Together with general modeling, they form the expertise of creating "well-formed" DSML.

Definition: A "well-formed" DSML is a metamodel created by a domain expert following the concepts of metamodeling, modeling and graphical language modeling.

When defining a DSML, a language designer starts by defining an abstract syntax i.e. a metamodel. When the abstract syntax is incompletely defined, the designer starts defining how abstract syntax elements are graphically represented inside the concrete syntax, simultaneously creating the concrete syntax and the mapping of correspondences. The process is repeated until the abstract syntax is completely defined.

Each abstract syntax element has limited possibilities of graphical representation by a DSML which we refer to *graphical roles* or *roles*. For instance, if a class of an abstract syntax can be graphically represented; it can either be represented as a node, or as a labeled link porting an attribute. Hence, all graphical roles of each abstract syntax element can be calculated. Out of all calculated graphical roles, a set of conforming graphical roles can be chosen for each abstract syntax element. We call this the graphical role election process (see Figure 4). For each chosen role, the corresponding concrete syntax element can be generated and mapped to the corresponding abstract syntax element. Note that the graphical information representing shapes, size, colors, etc. is not generated and the generated concrete syntax information should be afterwards manually complemented by a language designer. Together, the abstract syntax, the generated concrete syntax elements and the mappings between them, form a DSML. A graphical editor can be generated for the latter. This is the basic principle of the semi-automatic concrete syntax generator. Such process is described by Figure 4, where the rounded rectangles represent processes and the normal ones represent data.

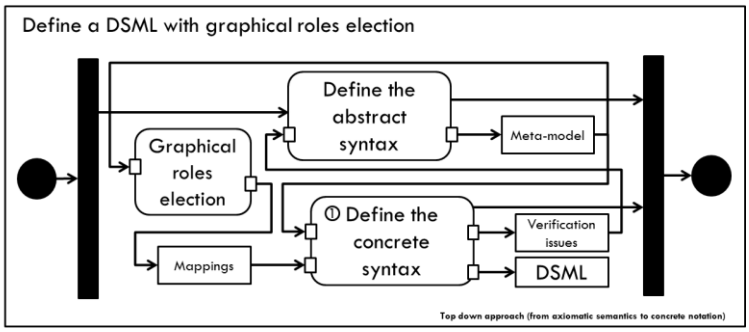


Figure 4. Defining graphical DSML with graphical roles election

In this section, we detail the graphical role election process as main part of the semi-automatic concrete syntax generator, having as foundation the previously described classifications. The whole process is divided into four sub-processes.

First, a point of view election is applied electing a class that can represent most classes. Then the role marking process is applied. This process marks the possible graphical roles of each abstract syntax element. Afterwards, a metamodel search is applied, detecting a set of labeled relationships (class-association pattern). Elected roles of each labeled relationship class are filtered and roles representing labeled relationship are associated. Finally, the conforming concrete syntax elements are generated for each graphical role and then associated to each abstract syntax element.

4.1. POV election

The first part of the automatic concrete syntax generation process is the POV election. Each metamodel may have as many POVs as classes it contains. We choose as a language canvas, the POV containing most classes. Therefore, a metamodel search is first applied on the abstract syntax, in order to calculate the number of classes, each class can represent if elected as a POV. We apply such process on the metamodel described on Figure 2. The results are shown in Table 2. It is then clear why the class *PersonalComputer* is elected as the POV of the language.

Table 2. POV election result.

Point of view	Containing classes	Total containing
PersonalComputer	Peripheral, PeripheralElement, Case, ConnectionRole, Motherboard, MotherboardElement	6
Peripheral	PeripheralElement, ConnectionRole	2
PeripheralElement	ConnectionRole	1
Case	Motherboard, MotherboardElement	2
ConnectionRole	/	0
Motherboard	MotherboardElement	1
MotherboardElement	/	0
Factory	/	0
NamedElement	/	0

4.2. Role marking

The role marking process begins by a metamodel search, divides first graphically representable classes and relationships from graphically not representable classes and relationships. Each representable class is marked as represented class or node.

We do not mark representable relationships. They are used to supplement the classes that they relate with more graphical roles i.e. *container*, *content* or *source*, *target*. For instance, the composition relationship *peripheralElements* between the classes *Peripheral and PeripheralElement* will assign a graphical role of *container* to the class *Peripheral* and a graphical role of *content* to the class *PeripheralElement*. Such roles might afterwards be translated as a graphical embedding. Second, not representable classes are visited in order to distinguish generic classes from language classes. Here again, a metamodel search is applied verifying if a given not-representable class subsumes representable classes. In that case, the visited class is marked as a generic class.

We present this process on the metamodel shown by Figure 2. First, representable elements and not representable elements are marked as shown in Table 1. Afterwards, graphical roles are associated to each class as described by Table 3.

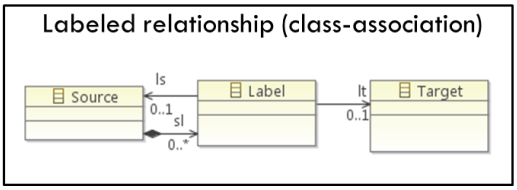


Figure5. Labeled relationship.

Table 1. Graphical roles election.

Class	Roles				
	Type	Source	Target	Container	Content
Peripheral	Node	no	no	peripheralElements	peripheral
Case	Node	no	no	motherboard	case
PeripheralElement	Node	no	from	role	peripheralElements
ConnectionRole	Node	to, from	no	no	role
Motherboard	Node	no	no	motherboardElements	motherboard
MotherboardElement	Node	relatedTo	relatedTo	no	motherboardElements
Factory	Language class	/	/	/	/
NamedElement	Generic class	no	no	no	no

4.3. Labeled relationships marking and role filtering

This phase consists of detecting labeled relationships (see Figure 5) of a given language. A metamodel search is affected on the metamodel identifying each possible class that can be represented as a label of such relationship (e.g. class Label of Figure 5). Such classes are always in relationship with two other classes, considered as source and target. The role marking phase has already assigned each class of the metamodel with graphical roles. Therefore, the roles associated to the labeled relationships elements should be regenerated taking into account this relationship. We continue the demonstration on the metamodel of Figure 2. Only the class *ConnectionRole* is detected as a label of a labeled relationship. The previously identified roles of this class (see Table 3) are deleted and a graphical role of labeled link is associated. The class *PeripheralElement* is filtered from the graphical roles target (*from* relationship) and container (*role* relationship), and assigned with the role source of the labeled relationship. The class *Case* is also filtered from the role target (*to* relationship) and assigned as target of the labeled relationship. Note that the rest of the graphical roles of a source and target classes of the labeled relationship are not filtered. For instance, the graphical role container (*motherboard* relationship) of the class *Case* is not filtered.

4.4. Automatic generation

Once all graphical roles are calculated and filtered, a set of conforming concrete syntax information can be generated, representing each calculated role. First, generic classes are complemented with graphical roles representing each property of the generic element. Finally, each graphical role is generated (concrete syntax) and associated to the corresponding class (correspondence mapping). Such process generates the concrete syntax information shown in Table 4 and Table 5 and maps it to corresponding abstract syntax elements.

Table 2. Automatically generated concrete syntax (nodes).

POV (point of view)	*TLN* (top level nodes)	*Child nodes*
PersonalComputer	Peripheral, Case	PeripheralElement, Motherboard, MotherboardElement

Furthermore, the language designer should manually complete the concrete syntax and the mapping of correspondences, either by adding supplementary design decision, or by modifying the already auto-generated one. Finally, either a graphical DSML is created and a graphical editor can be generated, or a validation issue is detected which repeats the whole process (see Figure 1).

Table 3. Automatically generated concrete syntax (links).

Embedding links	Relating links	Labeled links	Generic properties
Peripheral – PeripheralElement, Case – Motherboard, Motherboard – MotherboardElement,	MotherboardElement – MotherboardElement	PeripheralElement - Case	name

Figure 6 is an example of PC model conforming the PC language shown by Figure 2. The model is created by a graphical editor generated by the framework Diagraph.

5. Conclusion and perspectives

This paper has introduced and demonstrated a concrete syntaxes generator assistant for domain specific modeling languages. We demonstrated concepts and mechanisms allowing to guide and to assist a domain expert in order to define and formalize the concrete syntax of a graphical DSML. This work described how graphical DSML elements can be classified based on abstract syntax elements and on concrete syntax elements. A general typology for graphical DSML elements is described presenting the different elements that a graphical DSML can represent by a graphical editor. Furthermore, based on the classifications, we presented how a part of the concrete syntax can be generated automatically from an abstract syntax by a graphical role election.

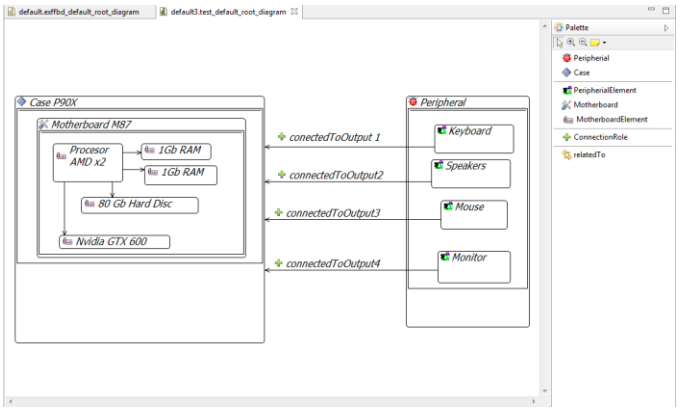


Figure 6. A model of PC conforming to the PC language.

This works open new perspectives. We aim to mathematically formalize the previously presented classifications. We also aim to provide a set of OCL constraints representing our classifications. Such constraints can guide the language

designer in the process of creating the abstract syntax of a graphical language. We believe that such verification will improve the way of creating graphical DSML.

References

Cho, H., 2011. A demonstration-based approach for designing domain-specific modeling languages. ACM, pp. 51–54.

Chomsky, N., 1965. *Aspects of the Theory of Syntax* Anonymous, ed., MIT Press.

Clark, T., Sammut, P. & Willans, J., 2008. Applied metamodelling: a foundation for language driven development.

Fuentes-Fernández, L. & Vallecillo-Moreno, A., 2004. An Introduction to UML Profiles. *European Journal for the Informatics Professional*, V(2), pp.6–13.

Gronback, R.C., 2009. *Eclipse Modeling Project: A Domain-Specific Language (DSL) Toolkit*, Addison-Wesley Professional.

Jouault, F. & Bézivin, J., 2006. KM3: a DSL for Metamodel Specification R. Gorrieri & H. Wehrheim, eds. *Lecture Notes in Computer Science*, 4037, pp.171–185.

Kent, S., 2002. Model Driven Engineering M. Butler, L. Petre, & K. Sere, eds. *Integrated Formal Methods*, 2335(2), pp.286–298.

Kleppe, A., 2007. A Language Description is More than a Metamodel. *Syntax*, (612), pp.1–9.

Kolovos, D.S. et al., 2010. Taming EMF and GMF Using Model Transformation. In *Proceedings of the 13th International Conference on Model Driven Engineering Languages and Systems: Part I*. Berlin, Heidelberg: Springer-Verlag, pp. 211–225.

Ledeczi, A. et al., 2001. The Generic Modeling Environment. In A. Ledeczi et al., eds. *Meta*. IEEE, pp. 1–14.

Mellor, S.J. & Balcer, M.J., 2002. *Executable UML: A Foundation for Model-Driven Architecture*, Addison-Wesley Professional.

MOF, O., 2002. OMG Meta Object Facility (MOF) Specification v1. 4. , (April).

Moody, D.L., 2009. The "Physics" of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering. *IEEE Transactions on Software Engineering*, 35(6), pp.756–779.

Omg, 2006. Meta Object Facility (MOF) Core Specification Omg, ed. *Management*, 080907(January), pp.1–76.

OMG, 2011. *UML 2.4.1-Infrastructure Specification*.

Pfister, F. et al., 2013. A light-weight annotation-based solution to design Domain Specific Graphical Modeling Languages. *ECMFA 2013, Montpellier, France, July 1-5, 2013.*.

Pontisso, N. & Chemouil, D., 2006. TOPCASED Combining Formal Methods with Model-Driven Engineering. *21st IEEE/ACM International Conference on Automated Software Engineering (ASE'06)*.

Specification, O.M.G.A. & Bars, C., 2007. OMG Unified Modeling Language (OMG UML). *Language*, (November), pp.1 – 212.

Steinberg, D. et al., 2008. *EMF: Eclipse Modeling Framework* E. Gamma, L. Nackman, & John Wiegand, eds., Addison-Wesley Professional.

Analyse OLAP d'un entrepôt de documents XML

Fatma Abdelhedi, Landry Ntsama, Gilles Zurfluh

IRIT-SIG, Université de Toulouse 1 Capitole
2 rue du Doyen Gabriel Marti, 31042 Toulouse, France
prenom.nom@irit.fr

RESUME. Les systèmes OLAP basés sur des entrepôts de données sont aujourd'hui bien intégrés dans les organisations, ils facilitent le traitement et l'analyse de l'information pour la prise de décision. Le développement du Web a conduit à l'accroissement du volume de données traité, ainsi qu'à la diversification des sources de l'information. Ce problème de diversification a été en partie résolu grâce au langage XML. Celui-ci permet en effet le traitement et l'échange de données complexes et hétérogènes. Seulement c'est un format qui s'adapte mal aux systèmes OLAP et d'entrepôts classiques. De plus il n'existe à ce jour aucun standard permettant de répondre à cette problématique. Aussi nous avons développé un modèle multidimensionnel qui utilise le formalisme orienté objet UML pour décrire un entrepôt de documents XML orientés-document. Le schéma de cet entrepôt (appelé StarCD) représente la structure des documents à analyser, telle qu'elle est connue par le décideur. Et dans cet article nous présentons un nouveau langage d'analyse OLAP destiné aux décideurs, qui permet d'exprimer des requêtes complexes sur un entrepôt de documents XML décrit par un StarCD.

ABSTRACT. OLAP systems based on data warehouses are nowadays well integrated into organizations and they ease the process of information analysis for decision-making. The Web expansion led to increase the volume of the data handled, as well as the sources diversification. This problem was partially solved thanks to the XML format, which indeed allows processing and exchanging complex and heterogeneous data. Only, this format does not fit for classic OLAP systems and warehouse, furthermore there is not an existing standard allowing solving this issue. So we developed a multidimensional model which uses the object oriented model UML to describe a XML Document Warehouse (XDW). The warehouse schema (named StarCD) represents the documents structure in the source, such as it is known by the decision-maker. And we present in this article a new OLAP analysis language intended for decision-makers, which allows them to express complex query on a XDW described by its StarCD.

MOTS-CLES : OLAP, Modélisation multidimensionnelle, XQuery, Entrepôt de documents XML.

KEYWORDS: OLAP Systems, XML Warehouse, XQuery, Multidimensional modeling

1. Introduction

Depuis une vingtaine d'années, les entrepôts de données et les systèmes OLAP associés ont été développés afin de répondre aux exigences de la prise de décision. L'évolution rapide des nouvelles technologies, notamment du Web, a fait naître de nouvelles problématiques liées à l'exploitation de données complexes et hétérogènes. Le format XML *(W3C, 2013a)* permet le traitement et l'échange de ce type de données à travers le Web. Aussi son exploitation dans les systèmes décisionnels nécessite la définition de nouveaux modèles *(Pérez et al, 2008)*.

L'intégration des documents XML dans les entrepôts a fait l'objet de nombreux travaux, dont la majeure partie concerne les documents XML « orienté-données » *(Pérez et al, 2008; Pokorny, 2001 ; Golfarelli et al, 2001)*. D'autres travaux ont traité de l'intégration des documents « orientés-document » *(Nassis et al, 2005 ; Tseng et Chou, 2006)*, en proposant de nouveaux modèles multidimensionnels ((par exemple le modèle en galaxie), et en étendant les langages d'analyse existant (SQL, MDX ou XQuery) pour l'analyse OLAP de ces entrepôts. Certains de ces modèles proposent de déstructurer le document afin de le représenter dans le schéma multidimensionnel, rendant ainsi la structure méconnaissable pour l'utilisateur. De plus, le décideur étant un non informaticien, la manipulation d'un langage comme XQuery *(W3C, 2013d)* dans le contexte d'un entrepôt XML peut s'avérer délicate.

Notre solution consiste à créer un entrepôt de documents XML (orientés-document) à partir d'une source contenant une collection de documents thématique (dont les structures sont unifiées). Pour l'élaboration du schéma multidimensionnel, nous avons étendu le modèle en étoile classique *(Golfarelli et al, 1998)*, les faits, dimensions et mesures sont extraits des XML-Schémas (XSchémas) *(W3C, 2013c)* décrivant les documents de la source. La structure hiérarchique des documents est donc préservée et facilite la compréhension du schéma multidimensionnel par les décideurs. Mais la contribution principale de cet article est la définition d'un langage d'analyse OLAP qui permet à des décideurs d'analyser un entrepôt de documents XML. C'est un langage dont la syntaxe relativement simple repose sur les principes développés dans les langages de requêtes pour objets complexes *(Barry et Cattell, 1997)*.

L'article est organisé comme suit. La section 2 présente un état de l'art relatif à nos travaux présentés. La section 3 présente une définition formelle du modèle multidimensionnel. La section 4 présente le langage d'analyse pour les décideurs. La section 5 présente le processus de traduction et la section 6 conclut cet article en soulevant des points de perspectives.

2. Etat de l'art

Dans *Ravat et al. (2007)*, les auteurs proposent un nouveau modèle multidimensionnel pour l'analyse OLAP des documents XML. Ce modèle dit « en galaxie » permet d'élaborer un schéma d'entrepôt sous la forme d'un graphe de

dimensions. Les dimensions d'une galaxie sont liées entre elles par un ou plusieurs nœuds exprimant la compatibilité entre les dimensions. C'est au moment de l'interrogation de l'entrepôt que le fait est désigné parmi les dimensions. Ainsi une dimension dans un schéma en galaxie représente à la fois une dimension et un sujet d'analyse. L'auteur étend l'algèbre multidimensionnelle classique qu'il adapte à son modèle en y intégrant de nouveaux opérateurs notamment pour l'analyse de données textuelles. Dans ces travaux, le modèle de données proposé pour l'entrepôt ne préserve pas la structure initiale des documents. Les constituants (éléments) des documents sont déstructurés (indépendants tout en restant liés) dans le schéma de l'entrepôt et l'on peut raisonnablement penser que ceci peut constituer un obstacle dans l'expression des requêtes par les décideurs.

Dans *Nassis et al. (2005)*, les auteurs proposent de décrire un entrepôt de documents XML en utilisant le modèle des diagrammes de classes d'UML. Dans le schéma de l'entrepôt, les documents sont représentés par un fait lié à des dimensions virtuelles. Les auteurs mettent l'accent sur l'aspect modélisation, sans définir d'approche concrète pour l'analyse OLAP via leur modèle. Ils présentent brièvement un algorithme de requête basé sur XQuery pour analyser un entrepôt. Le modèle de données proposé dans l'article permet de décrire un entrepôt de documents, en termes de faits et dimensions, grâce à des classes d'objets distinctes liées entre elles, et organisées suivant les besoins de l'utilisateur. Ainsi la structure des documents, telle quelle apparaît dans la source, n'est pas conservée dans l'entrepôt. Or cette structure est connue des décideurs qui souhaitent analyser des documents.

Et *Park et al. (2005)* proposent un modèle de données identique au précédent pour modéliser un entrepôt. Ils proposent d'étendre le langage MDX qui est à la base un langage de requête OLAP pour les entrepôts de données classiques. Ce langage étendu (XML-MDX) intègre une série d'opérateurs pour la manipulation d'un cube XML en permettant notamment l'analyse de contenus textuels. Le langage XQuery est utilisé lors la création du cube XML pour le calcul des mesures et la définition des dimensions. Tout comme dans l'approche présentée par *Nassis et al. (2005)*, la structure des documents à analyser ne se retrouve pas dans le schéma de l'entrepôt. De plus, le langage défini pour l'analyse OLAP s'avère complexe car il nécessite que le décideur ait des connaissances informatiques en manipulation de fichier XML (XQuery) ainsi qu'en analyse OLAP (MDX). Or celui-ci est par essence un non-informaticien.

Les approches présentées ci-dessus ne répondent que partiellement à notre problématique. Dans notre approche, nous supposons que les décideurs connaissent la structure des documents qu'ils désignent dans la source pour être analysés. Ils retrouveront cette structure dans le schéma de l'entrepôt ; et c'est cette structure qui leur permettra d'exprimer leurs requêtes d'analyse.

3. Définition du modèle multidimensionnel

3.1 Définition formelle

Le modèle que nous présentons dans cette section permet de décrire un entrepôt de documents XML sous la forme d'un schéma en étoile nommé StarCD. Ce modèle repose sur le formalisme UML *(OMG, 2013)* pour représenter le fait, ses dimensions et ses mesures. Il est élaboré à partir de l'analyse d'une source de documents XML décrits par un même schéma XML (XSchéma) et sert de support pour l'élaboration des requêtes d'analyse par les décideurs.

Un StarCD est défini à partir du XSchéma qui décrit les documents XML sous forme d'arbre :

StarCD = (*F, D*) où :

- F correspond au fait, c'est-à-dire à l'élément racine du XSchéma définissant la classe de documents à analyser.
- D = {D_1, …, D_n} est un ensemble de dimensions associé au fait.

Le fait est caractérisé par un ensemble de mesures, de types numérique ou textuel. Dans le StarCD, il représente l'ensemble des documents de la source à analyser. Les mesures sont décrites sous forme de classes liées au fait par des liens d'agrégation ; on retrouve ainsi la structure hiérarchique qui est présente dans le XSchéma de la source. Le fait est défini comme suit :

F = (*M, Agg*) où :

- M = {M_1, …, M_p} est un ensemble de mesures.
- Agg = {ag_1, …, ag_p} l'ensemble des fonctions d'agrégation associées aux mesures, avec *ag* dans {*SUM, COUNT, AVG, MAX, MIN*}.

Les dimensions sont quant à elles caractérisées par des ***paramètres*** organisés en hiérarchies. Ces hiérarchies précisent des niveaux de granularité allant du paramètre de plus haut niveau (All) au paramètre de plus bas niveau, celui-ci correspond à la classe liée au fait. Les dimensions sont définies comme suit :

D = {*P, H*} où :

- P = {p_1, …, p_s} est l'ensemble des paramètres de la dimension D
- H = {h_1, …, h_s} est l'ensemble des hiérarchies dans lesquelles sont organisés les paramètres, avec h défini comme suit :
    - h = {p_1, p_2, …, *All*} où p1 est le paramètre lié au fait (paramètre de plus bas niveau, et All étant le paramètre de plus haut niveau sur la hiérarchie.

Dans le StarCD, les dimensions sont décrites sous la forme d'un ensemble de classes ; chaque classe représente un paramètre qui permet de partitionner

l'ensemble des instances du fait. Chaque dimension est une arborescence de classes dont la racine est liée au fait par une relation d'association «By». C'est une relation d'association UML étendu pour indiquer l'axe choisi pour l'analyse. Les attributs faibles des dimensions correspondent aux attributs des classes.

Avec ce modèle nous introduisons le concept de ***hiérarchie inversée***. Dans ce type de hiérarchie, le paramètre de plus haute granularité est relié au fait pour respecter le XSchéma de la source. Le paramètre « All » considéré comme paramètre de plus haut niveau dans une hiérarchie « normale » n'est plus pris en compte ici.

3.2 Etude de cas

Pour illustrer notre démarche, nous disposons d'une collection d'articles scientifiques de même thématique que nous souhaitons analyser, et à partir de laquelle nous allons construire un entrepôt de documents. La source est décrite par un XSchéma (voir la Figure 1) représentant l'ensemble des documents, et la Figure 2 représente le StarCD qui en est extrait, contenant le fait, les dimensions et les mesures que nous avons sélectionnées pour cette analyse.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name = "Paper">
  <xs:complexType>
    <xs:sequence>
        <!--Types simples -->
        <xs:element name="Title" type="xs:string"/>

        <!--Types complexes -->
        <xs:element name = "CoAuthors">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="Author" maxOccurs="unbounded">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name = "Name" type="xs:string"/>
                        <xs:element name = "FirstName" type="xs:string"/>
                        <xs:element name = "Affiliation" type="xs:string" minOccurs="0"/>
                        <xs:element name = "Mail" type="xs:string" minOccurs="0"/>
                    </xs:sequence>
                    <!--Attributs auteur -->
                    <xs:attribute name = "H-Index" type="xs:integer" use ="required"/>
                  </xs:complexType>
              </xs:element>
              <xs:element name = "AffiliationGroup" type="xs:string" minOccurs="0"/>
          </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="Abstract" type="xs:string" minOccurs="0"/>
    <xs:element name="Résumé" type="xs:string" minOccurs="0"/>
    < !--    -->
```

Figure 1. XSchéma partiel de la classe de document Paper

Le fait et les mesures correspondent à l'arborescence de la racine Paper et sont situés dans la partie basse du StarCD. Les dimensions quant à elles figurent dans la partie haute. Comme pour les mesures, la structure hiérarchique des documents sources se retrouve dans la forêt des dimensions. Ceci contraint un sens particulier des liens hiérarchiques entre paramètres (***hiérarchie inversée***). Ainsi dans la Figure 1, la classe CoAuthors est directement liée au fait Paper parce que CoAuthors est un élément de premier niveau dans le XSchéma décrivant les documents sources ; les paramètres de cette dimension sont donc inversés par rapport au sens classique. De plus, d'après *Bordawekar et Lang (2005)* tout élément XML peut être analysé autant en tant que dimension que mesure. Ceci est dû à la classification entre les dimensions et les mesures qui n'est pas rigide, comme c'est le cas pour les données plates. Nous retrouvons cette propriété dans notre modèle, où chaque élément identifié comme dimension peut aussi être identifié comme mesure (par exemple la classe KeyWord dans la Figure 2). L'élément Date est néanmoins utilisé exclusivement comme dimension.

4 Langage d'analyse OLAP

4.1 Syntaxe et sémantique

Le langage de requêtes XQuery *(W3C, 2013d)* a été développé et standardisé par le W3C et il permet d'interroger des documents XML. Toutefois sa syntaxe est complexe et l'expression des requêtes parfois difficile. En effet, il nécessite une bonne connaissance d'une part du langage de sélection XPath *(W3C, 2013b)* permettant de naviguer dans la structure d'un document et d'autre part de la structure des fichiers XML. L'exemple 1 de requête suivant démontre la complexité du langage :

Exemple 1:

```
for $a in //DWordGroup/WordGroup,
    $b in distinct-values($a/KeyWord),
    $c in //DPubliDate/PubliDate,
    $d in $c/PubliMonth/PubliYear/@year
    let $doc := //Paper[WordGroup/@ref = $a/@id and
                        PubliDate/@ref = $c/@id]
return
   if (exists($doc))
   then <group>
                <KeyWord>{$b}</KeyWord>
                <PubliYear>{data($d)}</PubliYear>
                <val>{count($doc)}</val>   </group>
```

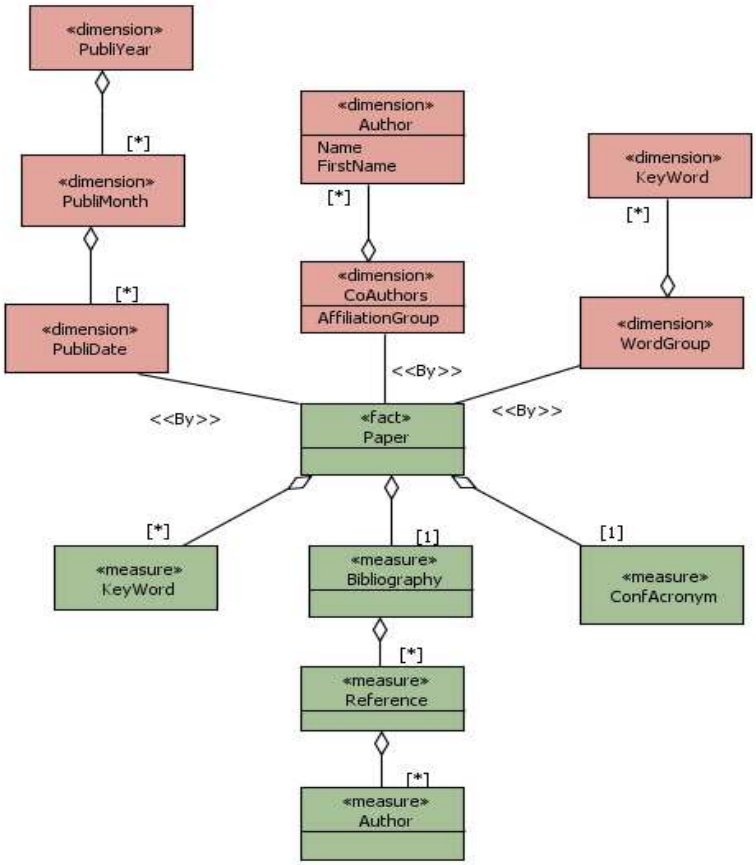


Figure 2. Diagramme en étoile StarCD

Aussi, le langage d'analyse présenté dans cette section permettra à des décideurs (non informaticiens) d'exprimer des requêtes complexes. Il s'inspire du langage OQL *(Barry et Cattell, 1997)* et permet de manipuler les objets représentés dans le schéma multidimensionnel StarCD. Sa structure facilite la navigation dans l'arborescence d'un fichier XML et sa syntaxe est de la forme suivante:

Analyse <*mesure*>

From <*variables désignant les classes de l'entrepôt*>

By <*dimensions*>

La clause Analyse contient les mesures ainsi que les fonctions d'agrégation qui leur sont appliquées. La clause From spécifie l'ensemble des éléments qui, au sein

du StarCD, sont utilisés soit comme mesure soit comme dimension ; chaque élément est associé à une variable de désignation (« alias ») qui permet d'éviter les ambiguïtés lors du parcours des hiérarchies. La clause By désigne les dimensions à utiliser comme axe d'analyse. La structure de la requête est définie suivant la grammaire BNF suivante :

<Query_spec> ::= <Analyse_clause><From_clause><By_clause>;

<Analyse_clause> ::= Analyse <Operator> '('<Measure_name>')'

<From_clause> ::= From <alias> in <XML_Element_Name>

(“.” <XML_Element_Name>)*(“,”<alias> in <XML_Element_Name>

(“.” <XML_Element_Name>)*)*

<By_clause> ::= By <alias>(',' <alias>)*

<AG_Fucnt> ::= COUNT | SUM | AVG | MAX | MIN

4.2 Opérateurs OLAP

Les auteurs de *(Ravat et al, 2006)* ont défini une série d'opérations algébriques, qui constitue un « noyau minimum fermé ». Dans cette série d'opérateurs, nous avons retenu les suivants :

- Opérateurs de forage : ***DrillDown***, ***RollUp***
- Opérateurs de rotation : ***DRotate***
- Opérateurs de structuration : ***Switch***, ***Nest***

Ces opérations ne sont pas explicitement définies dans notre langage, mais elles sont réalisables comme le montrent les exemples suivants.

Considérons la requête R1 définie à partir du StarCD de la figure 1. Cette requête calcule le nombre de mots clés pour chaque date de publication (paramètre PubliDate) :

R1: Analyse count(k)
From p in Paper,
k in p.KeyWord,
d in p.PubliDate
By d

4.2.1 Opérateurs de forage

Les opérateurs de forage permettent d'effectuer des analyses en agrégeant ou en désagrégeant les données. Ils procèdent à un changement de niveau de granularité soit vers le haut (RollUp) soit vers le bas (DrillDown).

Exemple 2:

RollUp : Cet opérateur agrège les mesures en changeant le paramètre d'une dimension. La requête R2 compte ainsi le nombre d'auteurs référencés par mois :

```
R2:  Analyse count(k)
     From p in Paper,
          k in p.KeyWord,
          d in p.PubliDate,
          m in d.PubliMonth
     By m
```

DrillDown : Cet opérateur est l'inverse du précédent ; il augmente le niveau de détail des mesures. Ceci équivaut à réduire le niveau de profondeur du paramètre utilisé pour le calcul dans la clause From. La requête R3 compte à nouveau le nombre d'auteurs référencés par jour :

```
R3:  Analyse count(k)
     From p in Paper,
          k in p.KeyWord,
          d in p.PubliDate
     By d
```

Dans le cas des hiérarchies inversées, l'application de ces opérateurs est aussi inversée. La progression dans la hiérarchie depuis le paramètre de plus bas niveau correspond à l'opération DrillDown.

4.2.2 Opérateurs de rotation

L'opérateur de rotation ***DRotate*** permet de changer l'axe d'analyse (la dimension) utilisée pour le calcul de la requête. Ceci équivaut à changer le paramètre voulu dans la clause From de la requête.

Exemple 3:

La requête R4 calcule le nombre de mots clés par groupe d'auteurs (CoAuthors). Elle effectue une rotation entre les dimensions PubliDate et CoAuthors depuis la requête R3 :

```
R3:  Analyse count(k)
```

From p in Paper,

k in p.KeyWord,

co in p.CoAthors

By co

4.2.3 Opérateurs de structuration

Ces opérateurs permettent d'organiser le résultat d'une requête. Ainsi, l'opérateur Switch, qui effectue un classement, permute les valeurs d'un paramètre de dimension affiché ; ceci équivaut à changer l'ordre des variables dans la clause By. D'autre part, l'opérateur Nest, qui réalise une imbrication, permet d'imbriquer un paramètre dans un autre.

Exemple 4:

Considérons la requête R5 ci-dessous qui calcule le nombre d'articles par auteurs, par date de publication et par mots-clés (paramètres Author, PubliDate, et KeyWord) :

R5: Analyse count(p)

From p in Paper,

k in p.WordGroup.KeyWord,

a in p.CoAuthors.Author,

d in p.PubliDate

By a.FirstName, a.LastName, k, d

L'opérateur ***Switch*** permet de changer l'ordre d'affichage des variables a.FirstName et a.LastName, la clause By devient donc :

By **a.LastName**, **a.FirstName**, k, d

L'opérateur ***Nest*** permet d'imbriquer la variable d dans la variable k ; la clause By devient donc :

By a.LastName, a.FirstName, **k.d**

Ces classements et imbrications n'ont qu'un impact visuel ; les valeurs affichées restent inchangées pour le décideur, seule la structure de l'affichage change. La requête R5 devient au final :

R5: Analyse count(p)

From p in Paper,

k in p.WordGroup.KeyWord,

a in p.CoAuthors.Author,

d in p.PubliDate

Considérant que la structure d'un fichier XML peut se représenter sous la forme d'un graphe arborescent, le langage que nous présentons permet de naviguer dans cet arbre, et aide le décideur à situer la profondeur des éléments lors de la formulation des requêtes d'analyse. Ces requêtes une fois exprimées sont traduites de façon transparente en XQuery par le biais d'un traducteur. Elles sont ensuite appliquées à l'entrepôt.

5 Expérimentation

5.1 Processus de traduction

La figure 2 représente l'architecture globale du traducteur que nous avons développé. Celui-ci assure la traduction automatique des requêtes en XQuery et leur application sur l'entrepôt ; il rend ce processus entièrement transparent pour le décideur. Il prend en entrée la requête OLAP écrite par le décideur et s'aide du schéma multidimensionnel pour construire la requête XQuery équivalente suivant le processus suivant :

(*1*) Une fois la requête saisie par le décideur, le traducteur procède dans un premier temps à une analyse syntaxique en vérifiant que les mots clés saisis sont bien ceux attendus et sont placés au bon endroit dans la requête.

(*2*) La requête est ensuite transmise au module générateur de code qui procède d'abord à une analyse sémantique (vérification du typage et de la conformité à la grammaire BNF), avant de procéder à la traduction de la requête en s'aidant des informations issues du schéma de l'entrepôt.

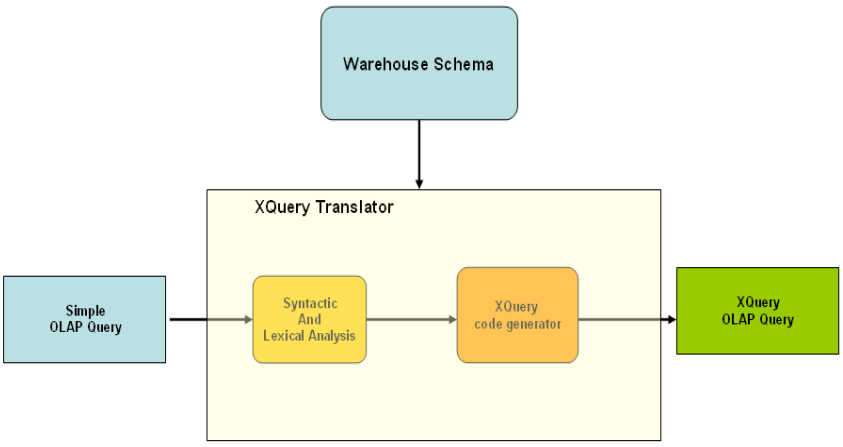


Figure 3. Architecture du traducteur

Nous avons défini une méthode de traduction qui peut se décrire informellement comme suit :

- ***Clause Analyse*** : Les mesures dans cette clause correspondent aux résultats à retourner. Elle seront déterminées dans le calcul de la clause ***let*** (qui permettra de déterminer les mesures à utiliser). Les mesures déterminées seront retrouvées dans la clause ***return*** du code XQuery accompagnées des fonctions d'agrégation associées. (Je ne comprends pas cette phrase).

- ***Clauses From*** : Pour chaque variable présente cette clause et aussi dans la clause **By**, il sera défini une clause ***for*** en XQuery, précisant elle aussi la profondeur des éléments ; le ***« distinct-values »*** en XQuery est utilisé si la variable n'a pas de fils dans l'arborescence.

  Chaque variable présente dans cette clause et aussi dans la clause Analyse, se retrouvera dans la clause ***return*** de la requête traduite.

- ***Clause By***: Les variables présentes dans cette clause ont plusieurs fonctions:

  - Elles permettent de préciser les dimensions (pour la définition des clauses ***for***)
  - Elles permettent de définir l'ordre d'affichage (clause ***order by***) si on travaille suivant plusieurs dimensions
  - Elles permettent d'appliquer une restriction sur les faits liés a la (aux) dimension(s) choisie(s) pour l'analyse (clause ***let***).

5.2 Comparaison de requêtes exprimées dans le nouveau langage et en XQuery

L'entrepôt matérialisé est alimenté à partir d'une collection d'articles scientifiques thématique, c'est à dire partageant la même structure (XSchéma). Il contient autant d'instances de fait que la source contient de documents. Les dimensions sont stockées dans des documents XML distincts, et sont liés au fait par des références comme on a pu s'en apercevoir dans la traduction de l'exemple 5. En effet nous avons séparé les instances de fait et les dimensions afin d'assurer l'unicité des valeurs des dimensions dans l'entrepôt, supprimant ainsi toute redondance.

L'entrepôt de documents est matérialisé dans la base de données eXist-DB qui est une base de données native XML, permettant une gestion aisée des collections XML par le biais du langage XQuery. Dans ce contexte, les requêtes exprimées par un décideur sont ensuite traduites en XQuery pour être appliquées à l'entrepôt, d'où l'importance du traducteur que nous avons développé. Nous montrons dans les exemples qui suivent quelques requêtes d'analyse ainsi que leurs traductions en XQuery réalisées par le traducteur :

Requête 1 : Calculer le nombre d'article publiés par auteur

Q1: Analyse count(p)

From p in Paper,
co in p.CoAuthors,
a in co.Author
By a.FirstName, a.LastName

Q1 traduite:

```
for $co in //DCoAuthors/CoAuthors,
    $a   in $co/Author
let $doc := //Paper[Dimensions/CoAuthors/ @ref=$co/@id]
return
<group>
      < AuthorFirstName >{$a/FirstName}</AuthorFirstName>
      < AuthorName >{$a/Name}</AuthorName>
      <val>{count($doc)}</val>
</group>
```

Requête 2 : Calculer le nombre moyen de mot clés utilisé par chaque auteur

Q2: Analyse avg(count(k))

From p in Paper,
k in p.KeyWord,
co in p.CoAuthors,
a in co.Author
By a.FirstName, a.LastName

Q2 traduite:

```
for $co in //DCoAuthors/CoAuthors,
    $a   in $co/Author
let $doc := //Paper[Dimensions/CoAuthors/ @ref=$co/@id]/Measures/KeyWord
return
<group>
      < AuthorFirstName >{$a/FirstName}</AuthorFirstName>
```

```
    < AuthorName>{$a/Name}</AuthorName>
    <val>{avg(count($doc))}</val>
</group>
```

Requête 3: Calculer le nombre d'article par mots clés et par années :

Q3: Analyse count(p)

```
    From p in Paper,
         w in p.WordGroup
         k in w.KeyWord
         m in p.PubliDate,
         y in m.PubliMonth.PubliYear
    By k, y
```

Q3 traduite:

```
for $w in //DWordGroup/WordGroup,
    $k in distinct-values($a/KeyWord),
    $m in //DPubliDate/PubliDate,
    $y in $c/PubliYear/@year

let $doc := //Paper[Dimensions/WordGroup/@ref = $a/@id
                    and Dimensions/PubliDate/@ref = $c/@id]
return
if (exists($doc))
then <group>
                <KeyWord>{$k}</KeyWord>
                <PubliYear>{data($d)}</PubliYear>
                <var>{count($doc)}</var>
   </group>
```

La condition if nous permet de filtrer sur les éléments $doc qui n'ont aucune valeur afin de ne pas surcharger l'affichage. Ces requêtes illustrent bien la simplicité d'expression d'une analyse OLAP dans notre langage par rapport à l'expression équivalente en XQuery. L'écart de forme de ces deux types de requêtes s'accroit avec la profondeur des éléments manipulés dans la structure hiérarchique des

documents. Par exemple, dans la requête 3, l'analyse se fait suivant deux dimensions. Dans la requête XQuery correspondante, on pourra constater la présence d'itérations marquées par la clause **for**. Cet aspect d'itération est simplifié dans le langage que nous proposons, grâce l'utilisation des variables définies dans la clause **From**. En effet celles-ci, par imbrication, permettent une navigabilité aussi aisée qu'en XQuery, tout en simplifiant l'expression de la requête. Le décideur bénéficie ainsi de toute la puissance de XQuery, en étant affranchi de sa complexité.

6 Conclusions et perspectives

Les systèmes OLAP actuels ne permettent pas le traitement et l'analyse des documents XML issus du Web. Or le format XML est devenu un standard pour le traitement et l'échange de données massives et hétérogènes.

Nous avons proposé dans cet article une approche de modélisation et d'analyse multidimensionnelle pour les entrepôts de documents XML natif. Nous avons défini un langage d'analyse OLAP pour les décideurs, qui s'applique sur un modèle multidimensionnel défini à l'aide du formalisme UML. Ce modèle permet de représenter la structure XML des documents à analyser à l'aide de faits, mesures et dimensions, facilitant l'expression des requêtes par les décideurs. Le langage permet aux décideurs de pouvoir exprimer des requêtes complexes à partir du schéma de l'entrepôt StarCD. Ces requêtes sont traduites en XQuery pour être appliquées sur un entrepôt de documents matérialisé. Nous avons à cet effet développé un module logiciel qui assure la traduction automatique des requêtes.

Nous développons actuellement un prototype basé sur l'approche définie dans cet article, et nous nous focalisons particulièrement sur les techniques d'intégration des données dans l'entrepôt à partir de la source des documents XML et du schéma (StarCD) de l'entrepôt. Nous cherchons également à optimiser le processus de traduction afin de gagner du temps sur le traitement des requêtes appliquées sur des masses de données et l'affinage des résultats. Il sera aussi question d'étudier notamment le temps de réponse lié à la traduction des requêtes. De même, l'adoption de ce langage passera probablement par le développement d'une interface homme-machine rendant plus ergonomiques, voire transparentes, les requêtes exprimées dans ce nouveau langage OLAP.

Références bibliographiques

Barry D. K., Cattell R.G.G. (1997). Titre du chapitre, *The Obkect Database Standard*, Morgan Kaufmann publisher, p. 83-115 (1997)

Beyer K., Chamberlin D., Colby L. S., Özcan F., Pirahesh H., Xu Y. (2005). Extending XQuery for Analytics, *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, p. 503-514

Golfarelli M., Rizzi S., Vrdoljak B (2001). Data warehouse design from XML sources, *Proceedings of the 4th ACM international workshop on Data warehousing and OLAP (DOLAP)*

Golfarelli M., Maio D., Rizzi S. (1998). The dimensional fact model: a conceptual model for data warehouses, *International Journal of Cooperative Information Systems 07*, 215-247.

Li Y. and An A. (2005). *Representing* UML Snowflake Diagram from Integrating XML Data Using XML Schema, *Proceedings of: 2005 International Workshop on Data Engineering Issues in E-Commerce (DEEC 2005)*, 9 April 2005, Tokyo, Japan

Nassis V., Rajagopalapillai R., Dillon T. S., Rahayu W. (2005). Conceptual and Systematic Design Approach for XML Document Warehouses, *Proceedings of the 2005 international conference on Computational Science and Its Applications*, p. 914-924.

Object Management Group (OMG): Unified Modeling Language (UML). *http://www.uml.org/*

Park B., Han H., and Song I. (2005). XML-OLAP: A Multidimensional Analysis Framework for XML Warehouses, *Data Warehousing and Knowledge Discovery*, Volume 3589, p. 32-42

Pérez J. M., Berlanga R., Aramburu M. J., and Pedersen T. B. (2008). Integrating Data Warehouses with Web Data: A Survey, Knowledge and Data Engineering, *IEEE Transactions on In Knowledge and Data Engineering*, IEEE Transactions on, Vol. 20, No. 7. (July 2008), p. 940-955

Pokorny J. (2001). Modelling Stars Using XML, *Proceedings of the 4th ACM international workshop on Data warehousing and OLAP (DOLAP)*

Rajesh Bordawekar and Christian A. Lang (2005). Analytical processing of XML documents: opportunities and challenges. SIGMOD Record, vol 34, pp. 27-32

Ravat F., Teste O., Tournier R., Zurfluh G. (2007). A Conceptual Model for Multidimensional Analysis of Documents, *Proceedings of the 26th international conference on Conceptual modelling*, p. 550-565

Ravat F., Teste O., Zurfluh G. (2006). Algèbre OLAP et langage graphique, *INFORSID*, p. 1039-1054 (2006)

Tseng, F. S. C., et A. Y. H. Chou (2006). The concept of document warehousing for multi-dimensional modelling of textual-based business intelligence. *Decision Support Systems (DSS)*, vol 42, Elsevier, pp. 727– 744.

Vrdoljak B., Banek M., and Rizzi S. (2003). Designing Web Warehouses from XML Schemas, *DaWaK 2003*, LNCS 2737, pp. 89-98

W3C Consortium (2013a). Extensible Mark-up Language (XML). *http://www.w3.org/XML/*

W3C-Consortium (2013b). Xml path language (Xpath) 3.0. http://www.w3.org/TR/xpath-30/

W3C-Consortium (2013c). Xml schema. http://www.w3.org/XML/Schema

W3C-Consortium (2013d). Xquery 3.0: An xml query language. http://www.w3.org/TR/xquery-30/

Session 4b

Systèmes d’information pour l’environnement

Évaluation de la vulnérabilité territoriale des enjeux environnementaux du Grand Lyon aux aléas technologiques.

SOTO Didier[1], RENARD Florent[1], MAGNON Audrey[1]

1. CRGA UMR 5600 EVS, Université Jean Moulin Lyon 3.
18, rue Chevreul. Boîte 20. 69362 Lyon Cedex 07, France.
didier.soto@univ-lyon3.fr

RESUME. Cette étude propose une méthodologie innovante, qui permet d'acquérir une connaissance précise de la vulnérabilité territoriale des enjeux environnementaux du Grand Lyon face au transport et au stockage de matières dangereuses. Celle-ci s'appuie notamment sur une compilation de données spatiales hétérogènes et un traitement statistique fondé sur des entretiens semi-dirigés d'experts et traduit, d'un point de vue cartographique, à l'aide d'un Système d'Information Géographique. Les résultats prennent la forme de représentations maillées à échelle fine, qui expriment, de manière homogène, la pondération de la concentration des enjeux selon leur sensibilité aux effets thermiques, toxiques et de surpression. En ce sens, les cartes produites peuvent être considérées comme des documents opérationnels pour la concertation des gestionnaires du risque, qu'ils soient industriels, acteurs des collectivités territoriales ou bien ingénieurs d'étude.

ABSTRACT. This study proposes a new methodology allowing a better knowledge of the territorial vulnerability of the Greater Lyon's environmental stakes to the transport and the storage of hazardous materials. This one is mainly based on a compilation of heterogeneous spatial data and a statistical processing, based on semi-structured interviews with experts and cartographically translated with a Geographical Information System. Here we propose fine scale raster views, which homogeneously weight the concentration of the environmental stakes according to their sensibility to thermal, toxic and overpressure effects. These maps can therefore be considered as operational documents for the mediation of industrials, local authorities 'officers and territorial engineers.

MOTS-CLES : enjeux environnementaux, SIG, vulnérabilité territoriale, données spatiales hétérogènes, information géographique homogène, analyse hiérarchique multicritère.

KEYWORDS: environmental stakes, GIS, territorial vulnerability, heterogeneous spatial data, homogeneous geographical information, analytic hierarchy process.

1. Introduction : des enjeux environnementaux non pris en compte dans les études de risque industriel.

De manière générale, la gestion du risque a longtemps privilégié la maîtrise de l'aléa, quel que soit son lieu d'application ou sa nature (Veyret et Reghezza, 2005 ; 2006). Or, depuis une vingtaine d'années, on assiste, d'un point de vue épistémologique, à un renversement de paradigme, si bien que les travaux actuels s'orientent de plus en plus vers une détermination des espaces exposés les plus vulnérables. L'appropriation du concept de vulnérabilité par les sciences humaines lui a conféré une dimension plus large, si bien qu'il est considéré aujourd'hui, dans la littérature, comme polysémique, multi-scalaire et multidimensionnel (Becerra, 2012).

Dans cette étude, nous nous sommes intéressés à un élément commun aux différentes vulnérabilités : le système territorial. Nous somme ainsi partis du principe qu'il existe, au sein de tout territoire, des enjeux d'ordres humains, environnementaux et matériels, qui permettent son fonctionnement et son développement. Or, si les enjeux humains et matériels sont prioritairement étudiés de par leur importance hautement stratégique (Leone, 2007), les enjeux environnementaux sont rarement pris en compte dans les études de risque. En France, les approches institutionnelles menées jusqu'à présent ont conduit à des modélisations spatiales qui présentent des limites pour la gestion territoriale des risques industriels. Ainsi, la méthode cartographique dite « déterministe », qui est employée dans la détermination des Plans Particuliers d'Intervention (PPI), génère une vision dichotomique du risque tandis que la méthode dite « probabiliste », qui est utilisée dans l'instruction des Plans de Prévention des Risques Technologiques (PPRT), ne permet pas de saisir concrètement la réalité du risque (Propeck-Zimmermann, 2010). Par ailleurs, un seul facteur de gravité est généralement pris en compte, la mortalité, au détriment des autres enjeux humains, matériels et environnementaux. Dans le cadre de la communauté urbaine du Grand Lyon, qui constitue notre terrain d'étude (figure 1), les documents officiels (Grand Lyon, 2004 ; 2010 ; Agence d'urbanisme pour le développement de l'agglomération lyonnaise, 2005 ; 2010) prennent très rarement en compte les enjeux environnementaux dans leurs études de risque. En outre, les échelles d'étude (commune, îlot de recensement IRIS de l'INSEE) sont insuffisantes et inadaptées aux contraintes opérationnelles du milieu, en raison de la diversité de forme et de taille du découpage administratif (Renard et Chapon, 2010).

Il s'avère cependant que les enjeux environnementaux constituent des éléments vulnérables, susceptibles d'impacter l'ensemble du territoire considéré, en cas de manifestation du risque (catastrophe). Un accident technologique, particulièrement lors d'un Transport de Matières Dangereuses (TMD), peut ainsi présenter un impact considérable sur l'environnement, occasionner des conséquences irréversibles (contamination toxique des ressources en eau) et générer des coûts importants pour la collectivité. Un tel événement est déjà survenu dans le Grand Lyon, sur la commune de Pierre-Bénite, le 7 décembre 2001, lorsqu'un accident de la circulation a causé le

déversement de 15 m³ d'hydrocarbures dans une station d'épuration par l'intermédiaire du collecteur d'eaux pluviales (base de données ARIA[1]).

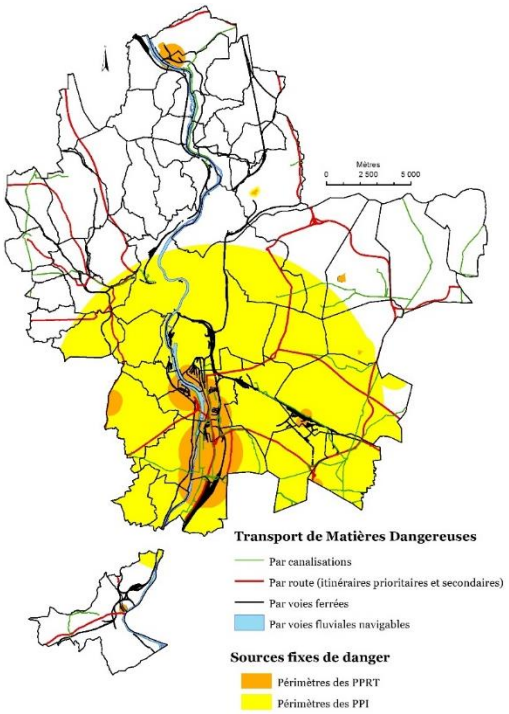


Figure 1. Périmètres et voies exposés à un accident dans le transport et ou/le stockage de matières dangereuses.

Sources : Grand Lyon, SPIRAL, base Prodige, VNF.
NB : les périmètres cartographiés correspondent au zonage des effets les plus majorants.

C'est précisément ce scénario que nous avons souhaité expérimenter au sein d'un territoire particulièrement vulnérable au stockage et au transport de Matières Dangereuses (MD). Actuellement, le Grand Lyon concentre 23 installations classées à risque « seuil haut » selon la directive Seveso II (figure 1), et 7 à risques « seuil bas ». Entre ces établissements s'organise une intense circulation de matières dangereuses, essentiellement par voies routières, mais aussi ferroviaires, fluviales, ou par l'intermédiaire de canalisations souterraines. Or, l'agglomération dispose de nombreux atouts environnementaux, comme en témoignent les 12 Projets Nature initiés par la communauté urbaine dans les coteaux de l'Ouest lyonnais, le long de ses

1. http://www.aria.developpement-durable.gouv.fr

axes fluviaux, ou dans les grandes plaines de l'Est. Le périmètre de l'un deux (Rhône aval-îles et lônes) entrecroise notamment ceux des PPRT de la Vallée de la Chimie.

2. Méthodologie : quantification et qualification de la vulnérabilité des enjeux environnementaux.

Au regard des éléments énoncés, il nous est apparu comme nécessaire de développer une méthodologie qui puisse permettre de quantifier et de qualifier la vulnérabilité des enjeux environnementaux à trois effets communs aux aléas technologiques : thermiques (incendie), toxiques (rejet accidentel d'une substance chimique) et de surpression (explosion). Pour cela, nous avons procédé en six étapes, détaillées ci-après. Il convient de préciser que les étapes 1, 2 et 6 ont nécessité de recourir à un Système d'Information Géographique (SIG), de manière à compiler, traiter et spatialiser les informations environnementales. Le logiciel utilisé pour cette étude est *ArcMap* 10, qui fait partie de la suite *ArcGIS* 10 (Service Packs 1 et 2), développée par la société ESRI.

Première étape : Elle consiste en l'élaboration d'un diagnostic du système territorial et en une recherche des enjeux potentiellement vulnérables à l'aléa stockage et transport de MD. Pour cette étude, nous avons choisi de procéder à une décomposition hiérarchique des enjeux environnementaux (tableau 1).

Enjeu rang 1	Enjeu rang 2	Origine des données	Spécificité des données	Facteurs de vulnérabilité
Environnementaux	Espaces agricoles	Grand Lyon	Espaces agricoles (S)	État des champs
	Espaces verts aménagés	Grand Lyon	Combinaison des espaces verts (S), des espaces récréatifs (S) et des espaces boisés (S)	
	Espaces naturels protégés + ZNIEFF	Grand Lyon ; base PRODIGE	Combinaison des espaces végétaux à préserver (S), des espaces boisés classés (S) puis intersection avec les arbres remarquables (P)	Avancement et types de culture
	Ressources en eau	Grand Lyon ; SAGE de l'Est Lyonnais	Combinaison des étangs (S), des plans d'eau (S), des mares (S) puis intersection avec l'hydrographie (L)	Type de bétail
	Zones de captage des eaux potables	Grand Lyon	Zones de protection rapprochées et éloignées des eaux potables (S)	
	Arbres d'alignement	Grand Lyon	Arbres d'alignement (P)	

Tableau 1. Décomposition hiérarchique des enjeux environnementaux.

NB : S : données surfaciques ; L : données linéaires ; P : données ponctuelles.

Une fois les différentes composantes identifiées, il s'est agi de construire une base de données spatiale, adaptée à notre problématique d'estimation de la vulnérabilité des enjeux environnementaux. Pour cela, nous avons pu bénéficier d'un accès au catalogue de données SIG du Grand Lyon (base Atlas), à partir duquel ont pu être extraites la majorité des informations. Il nous a fallu compléter notre compilation en téléchargeant des données en provenance d'autres bases, notamment la base PRODIGE Rhône-Alpes[2], qui consiste en un catalogue d'informations géographiques

2. http://www.georhonealpes.fr

qualitatives et quantitatives sur plusieurs thèmes (agriculture, air-climat, aménagement-urbanisme, culture-sports-sociétés-service). Les responsables du Schéma d'Aménagement et de Gestion des Eaux (SAGE) de l'Est lyonnais nous ont également transmis des données qualitatives relatives aux ressources en eau.

Deuxième étape : une fois les données relatives à chaque enjeu identifiées et compilées, il a été nécessaire de déterminer leurs interrelations hiérarchiques. Ici (tableau 1), notre typologie comporte six enjeux environnementaux de même rang hiérarchique : les espaces agricoles, les espaces naturels aménagés, les espaces naturels protégés et les Zones Naturelles d'Intérêt Ecologique Faunistique et Floristique (ZNIEFF), les ressources en eau, les zones de captage des eaux potables et les arbres d'alignement. Des facteurs de vulnérabilité sont également proposés, même s'ils ne sont pas exploités dans cette étude. Il s'agit ici d'une liste non exhaustive d'éléments (état des champs, avancement et types de cultures, type de bétail) qui permettent d'affiner la vulnérabilité des enjeux selon les effets considérés. En ce sens, ils peuvent être considérés comme des facteurs aggravant ou bien atténuant les impacts d'une catastrophe potentielle.

Dans chacune des grandes catégories d'enjeux présentées, il a été nécessaire de combiner des informations spatiales hétérogènes, aussi bien en termes de formes que d'emprise spatiale. Prenons l'exemple de la couche de données « espaces verts aménagés ». Celle-ci a nécessité de combiner trois types de données surfaciques différentes : les espaces verts (jardins publics entre autres), les espaces récréatifs (parcs de loisirs) et les espaces boisés. Pour créer une information homogène, il est nécessaire de procéder à une standardisation de la donnée spatiale puis à une pondération des cibles selon leur emprise spatiale sur le terrain (Kienberger *et al.*, 2009). Or, dans ce cas de figure, il ne s'agit pas d'une manipulation automatique à effectuer dans le SIG. Dans le cas du logiciel *ArcMap* 10, il a été nécessaire de créer un modèle de géo-traitement, à l'aide de l'application *ModelBuilder*. Le tutoriel du logiciel désigne par le terme de modèle une suite d'opérations (*workflow*), que ce dernier doit effectuer selon une programmation souhaitée par l'utilisateur en fonction de ses besoins. À titre d'exemple, la figure 2 présente les *workflows* qui ont été conçus pour procéder à la combinaison de trois jeux hétérogènes de données surfaciques en un carroyage homogène de valeurs quantifiées. Il a donc été nécessaire, dans un premier temps, de combiner les trois couches de données surfaciques, puis de procéder à une intersection et à une jointure spatiale avec un carroyage global du Grand Lyon, de 100 mètres de côté par mailles. S'ensuivent des opérations de géo-traitement pour déterminer la concentration de l'enjeu « espaces verts aménagés » dans chacune des 53 143 mailles du carroyage, en fonction de son emprise sur le terrain. La dernière opération consiste à rationaliser les résultats obtenus pour obtenir des valeurs comprises entre 0 et 1, la valeur 0 indiquant une absence de la cible dans la maille et la valeur 1 une concentration maximale. Le résultat final pour cet exemple est présenté

à la figure 3. Il convient de préciser que ce protocole n'est pas le même pour toutes les opérations de quantification des informations environnementales.

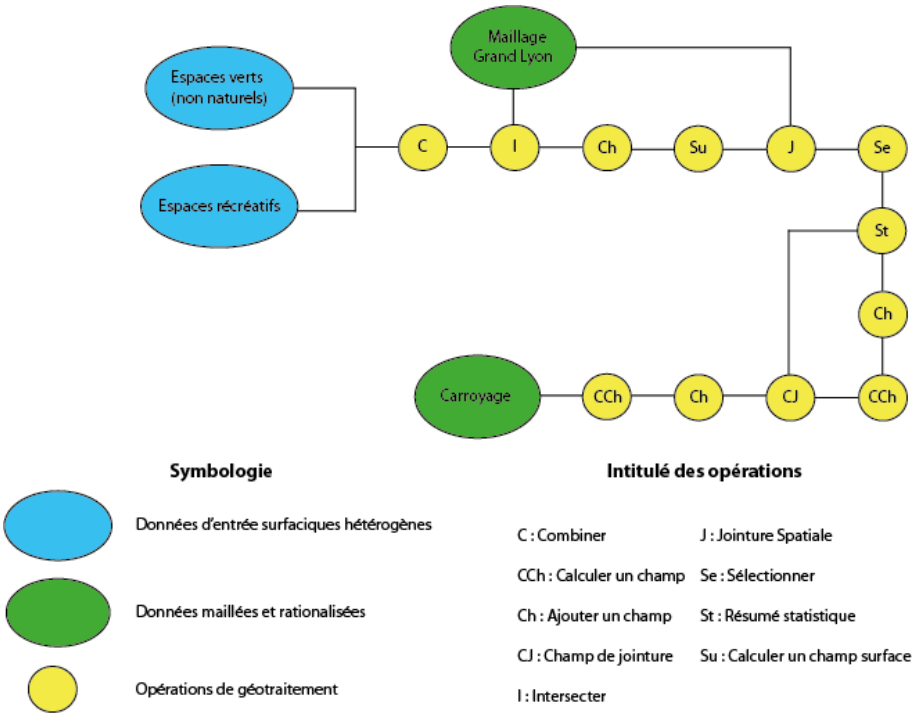
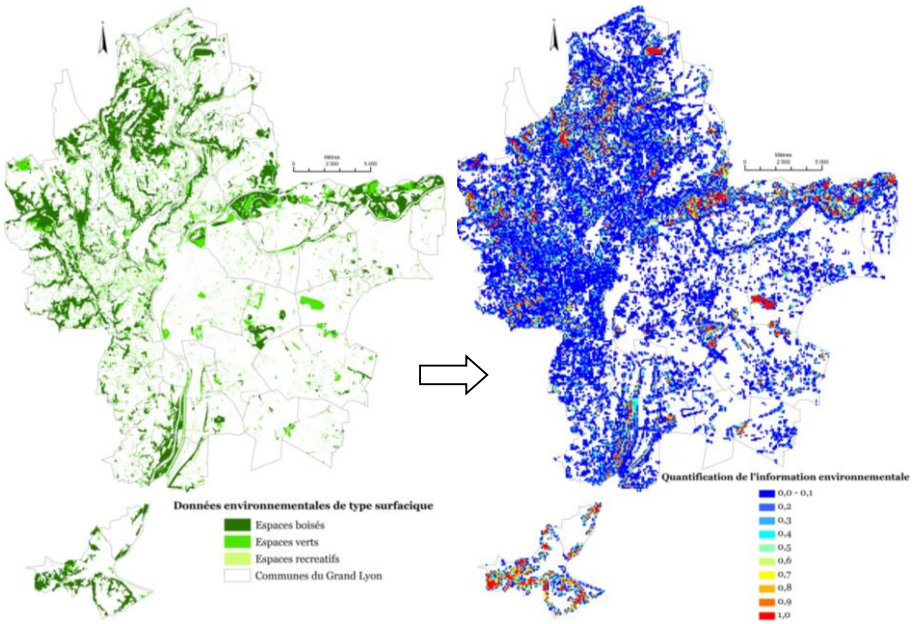


Figure 2. Workflows nécessaires à l'application ModelBuilder pour la combinaison de trois jeux hétérogènes de données surfaciques en un carroyage homogène de valeurs quantifiées.

Troisième étape : Une fois les enjeux quantifiés dans l'espace, il convient de définir des critères de pondération afin d'obtenir leurs fonctions de vulnérabilité. Cette étape repose donc sur l'évaluation de la vulnérabilité des enjeux de la décomposition hiérarchique à partir du jugement d'experts. Les méthodes de pondération sont multiples ; notre choix, justifié plus en détail lors d'une étude précédente (Renard et Chapon, 2010), s'est porté sur les méthodes multicritère, spécifiquement sur le processus d'analyse hiérarchique (*Analytic Hierarchy Process* – AHP) développé par Saaty (1980, 2008). Celui-ci autorise une approche systémique et déductive. Il est simple d'utilisation et permet de vérifier la cohérence et l'ensemble des jugements de comparaison.



Données hétérogènes (formes, emprise) *Carroyage homogène quantifié*

Figure 3. Traduction cartographique des opérations de géo-traitement de combinaison des données surfaciques hétérogènes, en ce qui concerne l'enjeu : « espaces verts aménagés ».

La problématique du choix des experts s'est avérée cruciale car les fonctions de vulnérabilité sont conditionnées par leurs jugements. La prise en compte de ce paramètre a nécessité l'établissement d'un panel représentatif des acteurs du risque industriel au sein du Grand Lyon, qui puisse mêler différentes représentations des vulnérabilités du territoire concerné. Pour cela, l'échantillon constitué a rassemblé des gestionnaires communaux du risque des trois principales communes exposées (Pierre-Bénite, Saint-Fons, Feyzin), des agents de la sécurité civile et de la prévention, des chargés d'études travaillant pour les services décentralisés de l'État (Directions Départementales des Territoires, Directions Régionales de l'Environnement, de l'Aménagement et du Logement), ainsi que des universitaires et des chercheurs de bureaux d'étude. Au total, 18 experts se sont prononcés lors d'entretiens semi-dirigés sur la vulnérabilité des cibles environnementales face aux effets thermiques, toxiques et de surpression provoqués par un éventuel accident lors du stockage et/ou du transport de MD (figure 4).

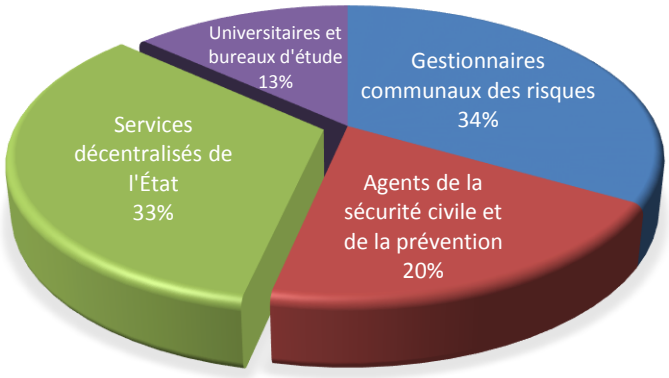


Figure 4. Distribution des experts du panel selon leurs caractéristiques professionnelles.

Chacun d'eux a ainsi été soumis à un questionnaire construit selon le principe d'une comparaison d'enjeux de même rang hiérarchique (tab.2). Les experts ont donc eu à évaluer la vulnérabilité de critères de même nature (ici, les enjeux environnementaux) en remplissant des matrices carrées d'ordre égal au nombre de cibles comparées. Dans l'exemple présenté, l'expert doit se prononcer sur la vulnérabilité relative des espaces agricoles par rapport aux espaces naturels aménagés. Pour cela, le curseur, placé sur la partie centrale de l'interface logicielle, doit être déplacé sur une échelle binaire de valeurs (1 à 9) pour indiquer un poids égal ou plus moins grand selon le positionnement de l'enjeu (tab.3). Cette opération est par la suite répétée pour chacune des lignes de la matrice.

Espaces agricoles	9 8 7 6 5 4 3 2 1 2 3 4 5 6 7 8 9	Espaces naturels aménagés

	Espaces agricoles	Espaces naturels aménagés	Espaces naturels protégés + ZNIEFF	Ressources en eau	Zones de captage des eaux potables	Arbres d'alignement
Espaces agricoles						
Espaces naturels aménagés						
Espaces naturels protégés + ZNIEFF						
Ressources en eau						
Zones de captage des eaux potables						
Arbres d'alignement						

Tableau 2. Exemple d'un questionnaire soumis aux experts lors des entretiens semi-dirigés (ici, la question porte sur l'évaluation de la vulnérabilité relative des espaces agricoles par rapport aux espaces naturels aménagés).

Pondération numérique	Pondération verbale
1	Vulnérabilité égale des deux éléments
3	Un élément est un peu plus vulnérable que l'autre
5	Un élément est plus vulnérable que l'autre
7	Un élément est beaucoup plus vulnérable que l'autre
9	Un élément est absolument plus vulnérable que l'autre
2, 4, 6, 8	Valeurs intermédiaires entre deux appréciations voisines
1/2, 1/3, 1/4, 1/5, 1/6, 1/7, 1/8, 1/9	Valeurs réciproques des appréciations précédentes

Tableau 3. Echelle de comparaison binaire utilisée pour évaluer la vulnérabilité territoriale (adaptée de Saaty, 1980).

Cependant, afin de valider la pertinence des réponses fournies par chaque expert, un ratio de cohérence doit être calculé à partir des valeurs propres des matrices de comparaison. La construction du questionnaire, de même que le calcul du ratio ont été réalisés, pour cette étude, à l'aide d'un logiciel spécifique : *Expert Choice* 11.5. Au terme des 18 entretiens conduits, ce ratio s'est révélé cohérent car inférieur ou égal à la valeur seuil de 0,2.

Quatrième étape : Les matrices de comparaison, renseignées à l'aide des jugements d'experts lors du stade précédent, ont permis au logiciel *Expert Choice* de calculer automatiquement le poids de chacun des éléments de la décomposition hiérarchique selon des valeurs comprises entre 0 et 1, le nombre 0 indiquant une vulnérabilité faible de la cible et le nombre 1 une vulnérabilité maximale. La figure 5 présente un graphique en secteurs des réponses communiquées par les experts sur la sensibilité des enjeux environnementaux à un effet toxique. Chacun des enjeux est ici pondéré (%) et, dans ce cas de figure, il s'avère que ce sont zones de captage des eaux potables qui constituent la cible la plus potentiellement affectée en cas d'effet toxique (33%).

Cinquième étape : Il est désormais nécessaire de procéder à l'agrégation des réponses collectées. Pour cela, la moyenne des jugements des différents experts est calculée par le logiciel, de manière automatique, afin de reconstituer une matrice unique de comparaison, qui va fournir les fonctions de vulnérabilité pour chacun des enjeux selon leur rang.

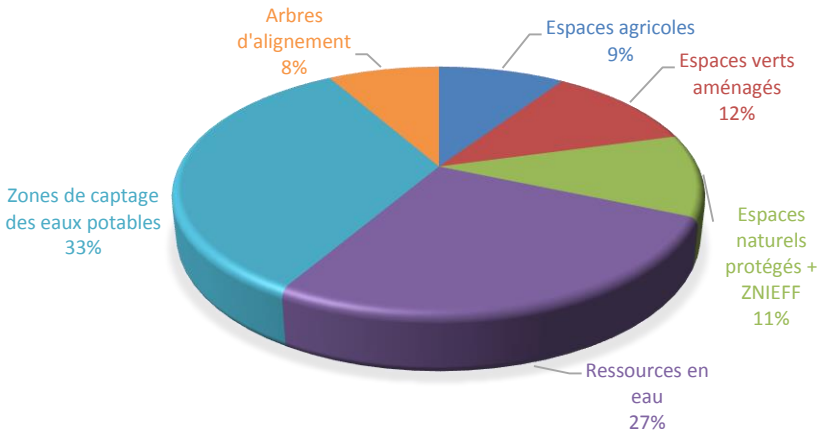


Figure 5. Pondération (%) des enjeux environnementaux, calculée par Expert Choice, à partir du jugement de l'ensemble des experts.

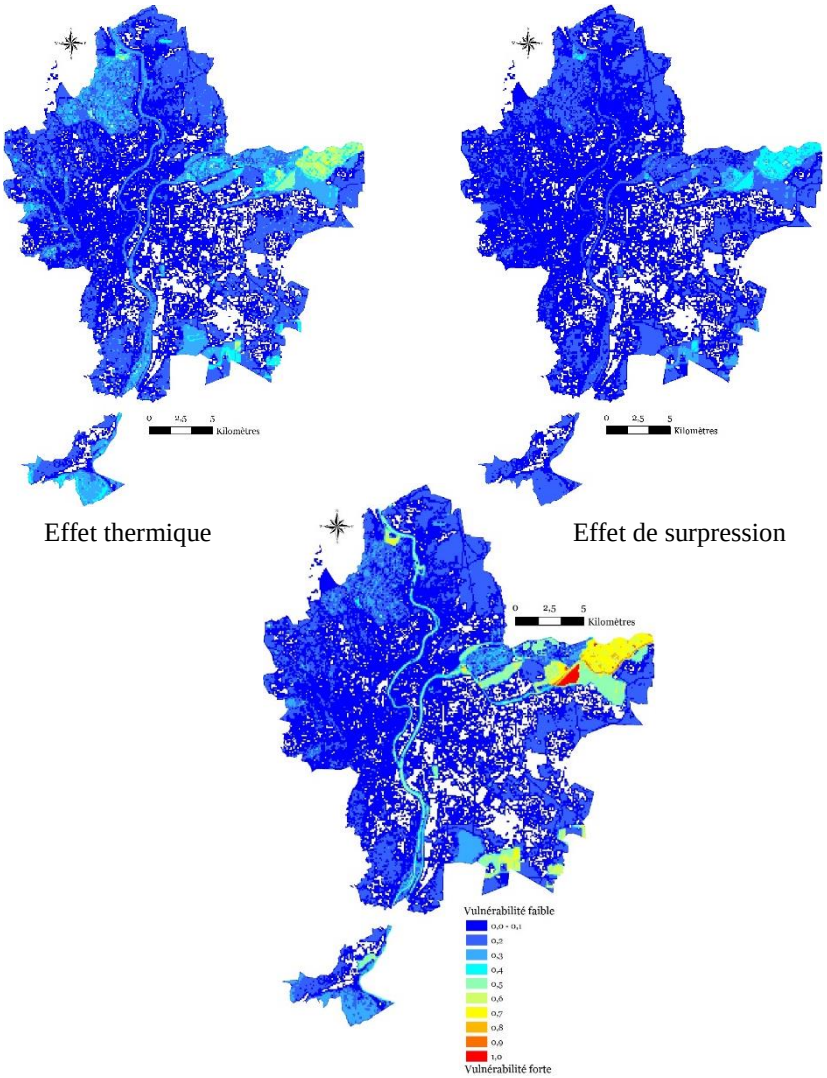
À ce stade, nous disposons donc d'informations environnementales standardisées et homogènes, obtenues lors de la seconde étape, et des valeurs de pondération pour chacune d'elles. La jointure de ces deux variables va être réalisée en recourant à nouveau aux fonctions de géo-traitement du logiciel *ArcMap* 10.

Sixième étape : La manipulation nécessaire pour produire les cartes finales de vulnérabilité des enjeux environnementaux (figure 6) consiste donc en un produit des coefficients de pondération avec les données de quantification spatiale de chacun des enjeux, rassemblés au sein d'une même table informatique par une opération de jointure attributaire. S'ensuit alors une jointure spatiale des résultats obtenus avec le carroyage global du Grand Lyon, afin d'obtenir une spatialisation standardisée et homogène des fonctions de vulnérabilité des cibles concernées.

3. Résultats : caractérisation des enjeux environnementaux les plus vulnérables de l'agglomération.

Les résultats de cette étude prennent la forme de supports cartographiques, déclinés selon chacun des effets considérés (figure 6). Les fonctions de vulnérabilité sont répertoriées ci-dessous (figure 7). Le niveau de résolution adopté dans cette étude permet une localisation précise des enjeux environnementaux présentant une sensibilité à un aléa TMD. L'examen des cartes, au regard de la figure 1, prouve que la vulnérabilité territoriale ne se limite pas uniquement aux zones exposées. La géographie des cibles les plus vulnérables pointe de manière assez nette le Nord-Est de l'agglomération, qui correspond aux anciennes plaines d'épandage du Rhône. C'est le cas notamment de la réserve artificielle du Grand Large, qui présente une vulnérabilité maximale aux effets toxiques. Or, celle-ci est localisée à proximité immédiate d'un axe prioritaire de TMD par route (N346, Rocade Est), qui traverse de

surcroît le Canal de Jonage. Le risque d'un événement similaire à celui du 7 décembre 2001 est probable, avec une dimension beaucoup plus majorante, compte tenu de l'emprise spatiale des ressources en eau et des zones de captage en eau potable. L'historique de l'accidentologie, conduit sur les trente dernières années dans le Grand Lyon, ne signale cependant aucune occurrence dans la zone dite du Grand Parc de Miribel-Jonage.



Effet thermique

Effet de surpression

Effet toxique

Figure 6. Vulnérabilité territoriale des enjeux environnementaux aux trois effets liés au transport et/ou au stockage de MD.

Vulnérabilité effet toxique (a)= 0.093 espaces agricoles+ 0.114 espaces verts aménagés + 0.108 espaces naturels protégés + 0.269 ressources en eau + 0.334 zones de captage des eaux potables+0.082 arbres d'alignement.
Vulnérabilité effet thermique (b)= 0.175 espaces agricoles+ 0.182 espaces verts aménagés + 0.22 espaces naturels protégés + 0.115 ressources en eau + 0.2 zones de captage des eaux potables+0.104 arbres d'alignement.
Vulnérabilité effet surpression (c)= 0.144 espaces agricoles+ 0.172 espaces verts aménagés + 0.18 espaces naturels protégés + 0.087 ressources en eau + 0.253 zones de captage des eaux potables+0.164 arbres d'alignement.

Figure 7 : Fonctions de vulnérabilité selon chacun des effets

Il est possible de faire le même constat en ce qui concerne le Sud-Est de l'agglomération, qui concentre également des zones de captage en eaux potables à proximité d'importants axes de circulations de MD (échangeur de l'A46 à hauteur de la commune de Mions, canalisation souterraine de gaz) et qui se trouve dans le périmètre d'étude du PPRT de l'entreprise Interra Log, localisée à Chaponnay (zonage M par rapport à un effet toxique). Ici, contrairement au Nord-Est de l'agglomération, un accident de TMD s'est déjà produit le 13 janvier 2006, lorsqu'un camion contenant 7 tonnes de chlorure de zinc s'est renversé, occasionnant la pollution d'un champ cultivé (base de données ARIA).

Les résultats obtenus permettent également de discriminer la vulnérabilité différente des enjeux environnementaux selon les effets (figures 6 et 7). D'un point de vue général, ces derniers sont particulièrement vulnérables aux effets toxiques, selon les jugements des experts rencontrés. Plus précisément, ce sont les zones de captage des eaux potables qui affichent la plus grande sensibilité par rapport à un effet toxique et/ou de surpression (respectivement 33% et 25%), tandis que, face à un effet thermique, les espaces naturels protégés et les ZNIEFF sont les plus durement impactés (22%). Les experts sollicités ont également pointé la spécificité des ressources en eau, qui sont beaucoup plus vulnérables à l'effet toxique (27%) qu'à un incendie (11%) ou à une explosion (8%). Cependant, si les ressources aquatiques et les zones de captage se démarquent nettement sur les cartes, les autres enjeux (espaces agricoles, espaces naturels aménagés) se caractérisent par leur plus faible vulnérabilité. Par conséquent, les Monts et les Coteaux du Lyonnais, ainsi que les Monts d'Or, situées à l'Ouest de l'agglomération et qui comportent des espaces agricoles (maraîchage) et d'importantes surfaces boisées (Agence d'Urbanisme pour le développement de l'agglomération lyonnaise, 2011), apparaissent comme potentiellement moins impactés.

4. Conclusions : vers une meilleure prise en considération des enjeux environnementaux dans les études de risque.

Ce travail d'évaluation de la vulnérabilité territoriale des enjeux environnementaux du Grand Lyon comporte trois intérêts principaux.

Le premier réside dans la constitution d'un SIG, qui a permis de compiler des données environnementales spatialisées, en provenance de diverses bases de données (Grand Lyon, PRODIGE, SAGE de l'Est lyonnais). Celles-ci, présentaient la particularité, à leur téléchargement, d'être hétérogènes, aussi bien en termes de formes (données surfaciques, linéaires et/ou ponctuelles) que d'emprise spatiale. Il a donc été nécessaire de procéder à une standardisation et une quantification des enjeux environnementaux considérés, à l'aide des modules de géo-traitement du logiciel utilisé. Le résultat de ces opérations géomatiques prend la forme d'une représentation carroyée de l'aire d'étude, qui permet de rendre homogène l'information souhaitée, en l'occurrence ici la vulnérabilité des enjeux environnementaux.

Le second est de permettre la géovisualisation d'un risque peu considéré dans les études institutionnelles et universitaires. La résolution spatiale des documents cartographiques permet de localiser précisément les enjeux les plus vulnérables. Dans le Grand Lyon, il apparaît que le Grand Parc de Miribel-Jonage concentre les cibles les plus vulnérables (ressources en eau, zones de captage des eaux potables), l'exposant à un accident lors du transport de matières dangereuses par route. Il est également possible de pointer une autre zone de danger environnemental au Sud-Est de l'agglomération à hauteur de l'échangeur autoroutier de Mions. En ce sens, les cartes produites peuvent être considérées comme des documents opérationnels pour la concertation des gestionnaires du risque, qu'ils soient élus ou bien ingénieurs territoriaux.

Le dernier intérêt de cette étude est de permettre une discrimination spatiale et fonctionnelle des effets relatifs aux aléas technologiques sur l'environnement, sur la base des jugements d'experts. La comparaison des cartes finales montre une vulnérabilité plus nette des cibles étudiées en fonction des effets toxiques potentiels, notamment dans le cas de fumées provenant d'un incendie, d'un nuage toxique ou d'une pollution accidentelle des sols.

Bibliographie

Agence d'urbanisme pour le développement de l'agglomération lyonnaise (2005). *Atlas des risques technologiques et de la vulnérabilité de l'agglomération lyonnaise*. Juin 2005.

Agence d'urbanisme pour le développement de l'agglomération lyonnaise (2010). *Indicateurs d'exposition des populations du Grand Lyon aux risques naturels et technologiques*. Observatoire du développement durable, juillet 2010.

Agence d'urbanisme pour le développement de l'agglomération lyonnaise (2011). *Les espaces naturels et agricoles de l'aire métropolitaine lyonnaise*. Représentations métropolitaines (carte au 1 :200 000ème), octobre 2011.

Becerra S. (2012). *Vertigo,* http://vertigo.revues.org/11988

Grand Lyon (2004). *Cahier risques majeurs du référentiel environnemental du Grand Lyon*.

Grand Lyon (2010). *Rapport de présentation – État initial de l'environnement – SCOT de l'agglomération lyonnaise*. Décembre 2010.

Kienberger S., Lang S., Zeil P., (2009). Spatial vulnerability units – expert based spatial modelling of socio-economic vulnerability in the Salzach catchment, Austria. *Natural Hazards and Earth System Sciences*, vol.9, p.767-778.

Leone F. (2007). *Caractérisation des vulnérabilités aux « catastrophes naturelles » : contribution à une évaluation géographique multirisque (mouvements de terrain, séismes, tsunamis, éruptions volcaniques, cyclones)*. Mémoire d'Habilitation à Diriger des Recherches, Université Montpellier 3.

Propeck-Zimmermann E. (2010). Caractériser les enjeux et les vulnérabilités : de l'analyse spatiale à un mode de représentation adapté à la concertation. *Actes du séminaire La prévention des risques industriels en France 2007 – 2009*, Paris.

Renard F., Chapon P.M., (2010). Une méthode d'évaluation de la vulnérabilité urbaine appliquée à l'agglomération lyonnaise. *L'espace géographique*, vol.39, n°1, p.35-50.

Saaty T.L. (1980). *The Analytic Hierarchy Process: Planning, Priority, Setting, Resource Allocation*, McGraw-Hill, New-York.

Saaty T.L. (2008). Relative measurement and its generalization in decision making. Why pairwiste comparisons are central in mathematics for the measurement of intangible factors. The analytic/network process. *Review of the Royal Spanish Academy of Sciences, series A, Mathematics*, vol.102, n°2, p.251-318.

Veyret Y., Reghezza M., (2005). Aléas et risques dans l'analyse géographique. *Annales des mines*, vol. 40, p. 61-69.

Veyret Y., Reghezza M., (2006). Vulnérabilité et risques. L'approche récente de la vulnérabilité. *Annales des mines*, vol. 43, p. 9-13.

Exploration de la factorisation d'un modèle de classes sous contrôle des acteurs

André Miralles[*] — Xavier Dolques[**] — Marianne Huchard[***] — Florence Le Ber[**] — Thérèse Libourel [****] — Clémentine Nebut[***] — Abdoulkhader Osman-Guédi[*,***]**

[*] *Tetis/IRSTEA, Montpellier, France, prénom.nom@teledetection.fr*
[**] *ICube, Univ. de Strasbourg/ENGEES, CNRS, Strasbourg, France*
[***] *LIRMM, Univ. Montpellier 2 et CNRS, Montpellier, France*
[****] *Espace Dev, Montpellier, France*
[*****] *Université de Djibouti, Djibouti*

RÉSUMÉ. *Nous nous intéressons à la construction du modèle de classes d'un système d'information environnemental. Cette construction est réalisée dans le cadre d'un processus de conception multi-acteurs dont les intérêts divergent ou sont conflictuels et qui nécessite de travailler en groupes autour d'un modèle qu'on essaie d'améliorer progressivement. Cette amélioration progressive se fait notamment en étudiant les classes et les associations et en faisant émerger des classes et des associations d'un niveau d'abstraction plus élevé. La technique que nous utilisons est dérivée de l'Analyse Formelle de Concepts. La construction de toutes les abstractions en une seule étape produit des nouvelles classes et des nouvelles associations en nombre trop élevé pour l'analyse par les experts. Nous proposons dans cet article de réaliser une construction par étapes, sous le contrôle des acteurs, de ces nouvelles classes et associations. Cette solution est testée sur un modèle de système d'information environnemental.*

ABSTRACT. *We are involved in the building of the class model of an information system in the environment domain. This building is done in the framework of a many-actors design process. Actor concerns are various and sometimes conflicting, and progressively designing the model during working group sessions is required. This progressive improvement relies on the emergence of more abstract classes and associations. We use an optimization technique based on Formal Concept Analysis. Nevertheless, building all abstractions in a single step produces too much concepts to be analyzed by the domain experts. In this paper, we propose a step-by-step approach, under the actor control, to extract these new abstractions. This solution is experimented on an environmental information system.*

MOTS-CLÉS : *refactorisation, Analyse Formelle de Concepts, Pesticides*

KEYWORDS: *refactoring, Formal Concept Analysis, Pesticides*

1. Introduction

Dans les domaines d'intervention d'Irstea (environnement, territoire, biodiversité, etc.), le développement d'un système d'information suppose de capturer des connaissances de plus en plus complexes et en évolution impliquant différents types d'acteurs parmi lesquels des scientifiques. Dans ce contexte, l'analyse du système est une phase cruciale du développement affectant directement la qualité du système d'information car c'est au cours de cette phase que les concepts du domaine et les exigences des acteurs sont capturés.

En situation multi-acteurs, l'analyse du système est effectuée, soit avec l'ensemble des acteurs au cours de séances d'analyse, soit par groupes. La première méthode peut poser des problèmes car, lorsque certains acteurs ont des intérêts divergents, certains d'entre eux vont avoir tendance à orienter l'analyse à leur bénéfice ou à être minimaliste. Dans un cas comme dans l'autre, le résultat est que le modèle final sera affecté et ne répondra pas aux besoins. Sa qualité est altérée. Aussi, effectuer l'analyse avec des groupes d'acteurs homogènes va permettre d'éviter partiellement ces inconvénients. La difficulté pour le concepteur réside dans l'intégration des différents points de vue pour obtenir un modèle unique. Il est confronté à un problème d'intégration de schémas et notamment de duplication de concepts. Il n'est pas rare que des concepts majeurs soient dupliqués dans plusieurs schémas. Par exemple, dans le modèle SIE Pesticides (Miralles *et al.*, 2011 ; Vernier *et al.*, 2013), *Pesticide* est un concept majeur d'un sous-modèle *Activités Agricoles* car les agriculteurs utilisent les pesticides dans leur itinéraire technique. Il existera aussi dans un sous-modèle *Activités Métrologiques* car on retrouve des pesticides dans les échantillons d'eau prélevés dans les cours d'eau.

L'objectif des recherches en cours conduit à identifier automatiquement ces doublons (illustrés ici au niveau des classes mais qui apparaissent aussi au niveau des attributs, des rôles, des opérations ou des associations) afin de simplifier le modèle. Lors de l'identification des doublons, de nouvelles abstractions vont également émerger naturellement par factorisation de caractéristiques communes à des classes, des attributs, des rôles, des associations ou des opérations. Par exemple, la factorisation des attributs *TypeProduction* d'un éleveur et d'un viticulteur va faire émerger un nouveau concept *Agriculteur* généralisant *Éleveur* et *Viticulteur*.

Dans nos travaux, nous utilisons l'Analyse Relationnelle de Concepts (ARC), méthode d'analyse de données qui offre une solution exacte et unique pour la suppression des doublons. L'inconvénient de cette méthode est que le processus est combinatoire et peut générer un nombre de nouveaux concepts qu'il est humainement difficile, voire impossible, d'analyser (Guédi *et al.*, 2013).

Afin de maîtriser cette explosion combinatoire, nous proposons ici une approche de factorisation pas-à-pas qui introduit une partie seulement des nouvelles abstractions à chaque étape. Notre approche présente comme autre avantage de mettre sous contrôle des acteurs le processus de factorisation, ceux-ci ayant la possibilité de valider ou de rejeter les concepts émergeant au cours de l'étape. Nous parlerons de *chemin de*

factorisation pour désigner une succession d'étapes lors desquelles sont effectués des choix d'éléments et de relations pour avancer dans le processus. La Figure 1 illustre cette notion de *Chemin*.

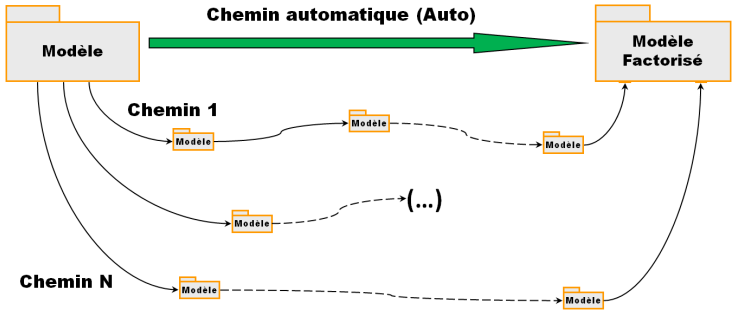


Figure 1. *Notion de chemins de factorisation*

La suite de l'article se présente de la manière suivante. Dans la section 2, nous présentons de manière succincte l'Analyse Relationnelle de Concepts appliquée aux modèles de classes et les problèmes identifiés dans diverses expériences menées précédemment. Dans la section 3, nous définissons la notion de chemin de factorisation. La section 4 est consacrée à la mise en œuvre de l'approche sur le modèle SIE Pesticides. Enfin nous présentons des travaux connexes dans la section 5, puis nous concluons dans la section 6.

2. Découverte d'abstraction par Analyse Relationnelle de Concepts

Une littérature assez riche montre comment la factorisation de modèles de classes articulés autour d'une hiérarchie de spécialisation/généralisation peut être mise en œuvre grâce à une méthode d'analyse de données basée sur la théorie des treillis (l'Analyse Formelle de Concepts (Ganter et Wille, 1999)). Une manière classique de l'appliquer consiste à décrire les classes par leurs caractéristiques (attributs et/ou opérations) dans une relation binaire (appelée contexte formel) (Godin et Mili, 1993). Les caractéristiques peuvent être décrites plus ou moins finement, et différentes formes de représentation dans le contexte formel peuvent être effectuées. De ce contexte formel peuvent être extraits des concepts, groupes maximaux de classes partageant des ensembles maximaux d'attributs et d'opérations. Ces concepts s'interprètent comme des classes, soit qu'ils représentent précisément une des classes de départ, soit qu'ils représentent une nouvelle super-classe introduite pour factoriser des caractéristiques communes. Les concepts sont organisés dans une structure de spécialisation appelée le treillis de concepts.

Dans la Figure 2(a), par exemple, l'attribut `name` se trouve dupliqué dans les classes `B1` et `B2` (Guédi *et al.*, 2013). Une méthode de factorisation basée sur l'AFC permettra d'obtenir le modèle de la Figure 2(b). Une nouvelle superclasse (désignée

par un expert `NamedElement`) est introduite automatiquement par l'AFC pour factoriser cet élément.

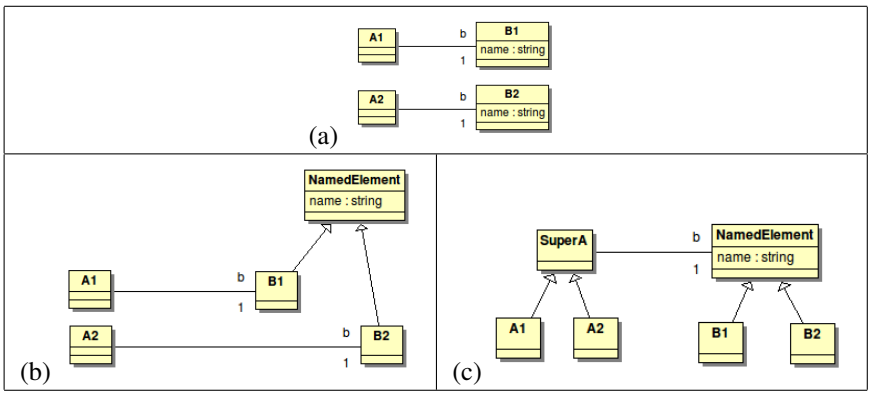


Figure 2. *Description du mécanisme de factorisation dans un modèle de classes*

Cependant les modèles de classes dans lesquels les attributs sont typés par des classes et dans lesquels on trouve également des associations ne peuvent être traités par ce modèle simple. Dans notre exemple, la méthode basée sur l'AFC ne calcule pas la factorisation plus complète présentée dans la Figure 2(c), dans laquelle une association généralise les deux associations de départ entre la nouvelle classe `SuperA` et la nouvelle classe `NamedElement`. Dans ce cas général, les données à traiter sont en effet constituées d'objets (classes, attributs, associations, rôles, opérations) décrits par des caractéristiques (leur nom, leur type s'il est primitif par exemple) mais également par des relations avec d'autres objets : une association possède des rôles, le rôle aboutit sur une classe, une classe possède un attribut, etc. L'Analyse Formelle de Concepts initiale a été étendue au cadre théorique de l'Analyse Relationnelle de Concepts (ARC, (Hacène *et al.*, 2013)) pour traiter de telles données et produire des factorisations plus systématiques et de portée plus grande comme celle de la Figure 2(c). Les données de l'ARC se présentent sous la forme d'une famille de contextes.

Définition 1. ***Famille relationnelle de contextes (RCF)***
Une famille relationnelle de contextes est un couple $(\mathbf{K}, \mathbf{R})$ *avec :*

– $\mathbf{K} = \{\mathcal{K}_i\}_{i=1,\ldots,n}$ *un ensemble de contextes formels* $\mathcal{K}_i = (G_i, M_i, I_i)$ *(relations objet-caractéristique), où* G_i *est l'ensemble des objets,* M_i *est l'ensemble des caractéristiques et* $I_i \subseteq G_i \times M_i$*.*

– $\mathbf{R} = \{r_j\}_{j=1,\ldots,m}$ *est un ensemble de relations objet-objet* r_j *où* $r_j \subseteq G_{i_1} \times G_{i_2}$ *pour* $i_1, i_2 \in \{1, \ldots, n\}$*.*

Dans le cadre de nos données, divers contextes formels et diverses relations objet-caractéristique peuvent être considérés. Par exemple une famille relationnelle de contextes peut se composer simplement de deux contextes formels respectivement pour les classes et les attributs ($\mathbf{K} = \{\mathcal{K}_{class}, \mathcal{K}_{attributs}\}$) et de deux relations ($\mathbf{R} =$

$\{r_{owns}, r_{hasType}\}$) indiquant quelle classe possède quel attribut et quel est le type d'un attribut. D'une famille relationnelle de contextes, l'ARC fait émerger itérativement des concepts. Lors de la première étape, un treillis de concepts est construit pour chaque contexte formel, par exemple un treillis de classes et un treillis d'attributs. Si les attributs sont décrits par leur nom dans le contexte formel $\mathcal{K}_{attribute}$, un concept de ce treillis groupe des attributs de même nom. Lors des étapes suivantes, les relations objet-objet sont intégrées sous la forme de caractéristiques relationnelles dans le contexte formel, en exploitant un opérateur de *scaling* (existentiel ou universel) et les concepts construits à l'étape précédente. Si une classe C est en relation avec un attribut a groupé dans un concept d'attributs C_a obtenu à l'étape précédente, on lui attribue la caractéristique relationnelle $\exists owns(C_a)$, pour indiquer que C possède l'un au moins des attributs groupés dans C_a.

Plusieurs expériences ont été menées par le passé, chacune s'appuyant sur une configuration (modèle de données) propre. Nous nous intéressons ici au point sensible qui a été relevé lors de ces expériences et qui est celui de la complexité pratique de l'approche. Pour simplifier, nous présentons seulement le nombre de concepts de classes car il s'agit de l'un des éléments de premier plan des modèles. Dans (Roume, 2004), on trouve une configuration assez complète dans laquelle sont pris en compte dans des contextes formels les classes, les attributs, les méthodes, les associations (décrites par des multiplicités), et les relations considérées sont association-classe, classe-attribut, attribut-classe et classe-méthode. Sur l'un des plus gros modèles étudiés (issu de France Télécom), formé de 57 classes, 167 concepts de classes apparaissent. Dans (Hacène, 2005), les contextes formels se limitent aux classes, propriétés (attributs et rôles) et les relations sont classe-propriété et propriété-classe. Avec cette description, sur un petit modèle de 6 classes (Jetsgo), 35 concepts de classes sont créés. Plus récemment, (Falleri *et al.*, 2008) ont utilisé une configuration identique et l'ont testée sur la librairie Apache Common Collections, où les 250 classes ont produit un nombre assez raisonnable de 284 concepts de classes. Sur le méta-modèle UML2, composé de 246 classes, l'approche produit le nombre fort élevé de 1780 concepts de classes. Ces quelques expériences, rapportées ici à grands traits, si elles montrent l'intérêt des concepts lors de l'analyse qualitative, font également ressortir les limites de l'approche. Les concepts de classes (ou des autres objets telles que les associations, les attributs, les opérations ou les rôles) issus de la méthode sont destinés à être examinés par des experts, et il semble difficile d'en présenter de tels nombres dans la plupart des cas. La section suivante s'attache à trouver une solution à ce problème.

3. Chemins de factorisation

Dans la partie précédente, nous avons montré comment l'ARC permettait de construire des formes normalisées de modèles de classes, mais également les limites qu'elle présente : une quantité potentiellement trop importante de nouvelles abstractions à présenter aux experts et des concepts extraits qui ne sont pas toujours utiles dans le domaine métier visé. Dans cette section, nous introduisons une approche exploratoire

appuyée sur l'ARC qui vise à réduire et à maîtriser ces limites. Nous proposons de contrôler l'ARC à différents niveaux, en nous arrêtant à chaque étape pour examiner quelles factorisations ont été réalisées et ainsi quelles abstractions sont apparues. A chaque étape, l'expert choisit les relations objet-objet et objet-caractéristique sur lesquelles il veut porter son attention, ainsi qu'un algorithme de construction associé à chaque table objet-caractéristique. La seule contrainte à ce choix est qu'une relation objet-objet ne peut être choisie que si une classification des objets de la cible de la relation a été calculée à une étape précédente (menant à la notion de chemin de factorisation valide que nous ne définirons pas ici pour des raisons de place). Deux sous-ordres des treillis vont être utilisés : les AOC-posets (Attribute-Object-Concept-posets) (Petersen, 2001), qui se définissent comme des sous-ordres des treillis restreints aux concepts introduisant un objet ou une caractéristique, et les icebergs(i), qui se limitent aux concepts couvrant un nombre d'objets supérieur à un seuil i choisi par un utilisateur.

Dans le contexte de la factorisation de modèles UML, l'opérateur de *scaling* est toujours existentiel. Pour en donner une intuition, prenons l'exemple de la relation qui associe à une classe ses attributs (`class_attribut`). Plusieurs attributs peuvent être groupés par exemple parce qu'ils portent le même nom. Quand on s'intéresse à la transformation de la relation `class_attribut` par l'opération de *scaling* et à son influence dans la création de groupes de classes, on ne s'intéresse ordinairement pas à trouver des groupes de classes qui ne vont partager que des attributs d'un groupe (par exemple que des attributs nommés "mesure", inférence qui serait réalisée par l'opérateur universel). On cherche plutôt à trouver des groupes de classes qui ont au moins un attribut du groupe (par exemple un attribut nommé "mesure", inférence qui serait réalisée par l'opérateur existentiel). Les choix réalisés successivement aux différentes étapes définissent ce que nous appellerons un chemin de factorisation. La notion de chemin de factorisation est inspirée de la notion de chemin d'exploration introduite pour l'utilisation de l'ARC dans le domaine de l'analyse de données (Dolques *et al.*, 2013).

Définition 2. ***Chemin de factorisation***
Pour une famille de contextes relationnels $(\mathbf{K}, \mathbf{R})$*, nous définissons une grammaire* (V_N, V_T, S, R) *pour décrire les chemins de factorisation.*
L'ensemble V_N *des non terminaux comprend les symboles entre '<' et '>'. L'ensemble* V_T *des terminaux est l'union des ensembles suivants : les entiers naturels* $\mathbb{N}$*, les relations objet-caractéristique* $\mathbf{K}$*, les relations objet-objet* $\mathbf{R}$*, les algorithmes* $\{AOC\text{-}poset, treillis, iceberg(i)\}$*, les opérateurs de scaling* $\{\exists, \forall\}$ *et les symboles* $\{\Rightarrow, [,], (,)\}$*. L'axiome* S *est* <$Chemin$>*. L'ensemble* R *des règles de production de* R *est le suivant :*

<$Chemin$> $\longrightarrow$ <$Etape$> | <$Etape$> <$Chemin$>
<$Etape$> $\longrightarrow$ <$NumeroEtape$> $\Rightarrow$ [<$DonneesEtape$>]
<$DonneesEtape$> $\longrightarrow$ <$PairesOC$> <$PairesOO$> |<$PairesOC$>
<$PairesOC$> $\longrightarrow$ <$PaireOC$> | <$PaireOC$> <$PairesOC$>
<$PaireOC$> $\longrightarrow$ <$RelationOC$> (<$Algorithme$>)
<$PairesOO$> $\longrightarrow$ <$PaireOO$> | <$PaireOO$> <$PairesOO$>

$<PaireOO> \longrightarrow <RelationOO> (<Operateur>)$
$<Algorithme> \longrightarrow AOC\text{-}poset \mid treillis \mid iceberg$
$<NumeroEtape> \longrightarrow i, i \in \mathbb{N}$ $<Operateur> \longrightarrow \exists \mid \forall$
$<RelationOC> \longrightarrow r, r \in \mathbf{K}$ $<RelationOO> \longrightarrow r, r \in \mathbf{R}$

Le chemin ci-dessous se compose d'une première étape où on ne considère que les classes et les attributs, classés dans un AOC-poset, et d'une seconde étape avec ces mêmes objets, classés dans un AOC-poset, ainsi que la relation qui associe à une classe ses attributs, avec l'opérateur de *scaling* existentiel :
$0 \Rightarrow [\mathcal{K}_{class}(AOC\text{-}poset)\ \mathcal{K}_{attribute}(AOC\text{-}poset)]$
$1 \Rightarrow [\mathcal{K}_{class}(AOC\text{-}poset)\ \mathcal{K}_{attribute}(AOC\text{-}poset)\ r_{class-attribute}(\exists)]$

4. Application sur le modèle SIE Pesticides

Dans cette section, nous décrivons la manière dont nous mettons en œuvre des chemins de factorisation pour le modèle SIE Pesticides. La section 4.1 décrit le méta-modèle utilisé, puis dans la section 4.2 nous définissons trois chemins de factorisation que nous comparons avec le chemin de factorisation classique (*Chemin Auto*) pour illustrer notre démarche. Ces quatre chemins sont ensuite analysés de manière qualitative sur un extrait du modèle dans la section 4.3. Dans la section 4.4, nous appliquons la même approche au modèle complet et nous étudions les résultats de manière quantitative afin de valider sa faisabilité pratique.

4.1. *Méta-modèle utilisé*

Le méta-modèle de données que nous avons choisi est présenté sur la Figure 3 (gauche). Les classes (contexte $\mathcal{K}_{class}$) y sont décrites par leur nom. On leur associe leurs attributs et leurs rôles (relations objet-objet $r_{class-attribute}$ et $r_{class-role}$). Les associations (contexte $\mathcal{K}_{association}$) sont décrites par leur nom. On leur associe leurs rôles (relation objet-objet $r_{association-role}$). Les attributs (contexte $\mathcal{K}_{attribute}$) sont considérés comme ayant des types primitifs (choix de modélisation dans le cadre de ce projet) et sont simplement décrits par leurs noms. Les rôles (contexte $\mathcal{K}_{role}$) sont décrits par leurs types qui sont des classes (relation objet-objet $r_{role-class}$). Les quelques opérations présentes (rares dans ce modèle destiné à l'analyse métier du système d'information) n'ont pas été considérées. Lorsque les rôles ne sont pas nommés, ils sont ignorés. Par ailleurs, lors de la conception du modèle de classes du SIE Pesticides, un soin particulier a été apporté à exprimer la navigation sur les associations. La Figure 3 (droite) présente un exemple de modèle composé de deux classes, munies d'attributs et de rôles dans une association (en haut) et sa traduction dans le méta-modèle choisi. Les noms des liens ne figurent pas car il n'y a pas d'ambiguïté sur leurs noms. On peut y noter que la classe $Person$ est connectée au rôle c, mais que la

classe Car n'est pas connectée au rôle p, pour traduire le sens de navigabilité. Cette représentation se rapproche de l'instanciation du méta-modèle UML classique.

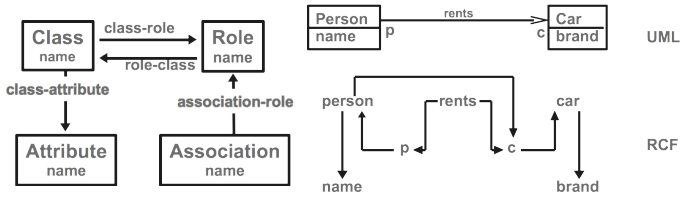


Figure 3. *Méta-modèle de données utilisé et modèle de classes (extrait) dans la représentation choisie*

4.2. *Chemins de factorisation*

Dans cette section, nous définissons quatre chemins de factorisation qui serviront à bâtir le cas d'étude. Le *Chemin Auto(matique)* (cf. Figure 1), utilisé jusqu'à présent par les utilisateurs de l'ARC, sera utilisé comme référence lors de l'analyse des résultats.

Chemin de factorisation. *Auto*

$0 \Rightarrow [\,\mathcal{K}_{class}(AOC\text{-}poset)\ \mathcal{K}_{attribute}(AOC\text{-}poset)\ \mathcal{K}_{role}(AOC\text{-}poset)\ \mathcal{K}_{association}(AOC\text{-}poset)]$

$i \Rightarrow [\mathcal{K}_{class}(AOC\text{-}poset)\ \mathcal{K}_{attribute}(AOC\text{-}poset)\ \mathcal{K}_{role}(AOC\text{-}poset)\ \mathcal{K}_{association}(AOC\text{-}poset)$

$r_{class-attribute}(\exists)\ r_{role-class}(\exists)\ r_{class-role}(\exists)\ r_{association-role}(\exists)],\ i \in [1,6]$

Le premier chemin commence par classer les classes et les attributs, puis les classes d'après leurs attributs, puis les rôles d'après les classes, les classes d'après leurs rôles et enfin les associations d'après leurs rôles. Il faut noter que, dans ce chemin, il y a une rupture à l'étape 2 à laquelle les classes, précédemment décrites par leurs attributs, ne sont plus décrites par aucune information.

Chemin de factorisation. *1*

$0 \Rightarrow [\mathcal{K}_{class}(AOC\text{-}poset)\ \mathcal{K}_{attribute}(AOC\text{-}poset)]$

$1 \Rightarrow [\,\mathcal{K}_{class}(AOC\text{-}poset)\ \mathcal{K}_{attribute}(AOC\text{-}poset)\ r_{class-attribute}(\exists)]$

$2 \Rightarrow [\,\mathcal{K}_{role}(AOC\text{-}poset)\ \mathcal{K}_{class}(AOC\text{-}poset)\ r_{role-class}(\exists)]$

$3 \Rightarrow [\,\mathcal{K}_{class}(AOC\text{-}poset)\ \mathcal{K}_{role}(AOC\text{-}poset)\ r_{class-role}(\exists)]$

$4 \Rightarrow [\,\mathcal{K}_{association}(AOC\text{-}poset)\ \mathcal{K}_{role}(AOC\text{-}poset)\ r_{association-role}(\exists)]$

Dans les chemins 2 et 3, on cumule le long du chemin les abstractions apprises, la définition du chemin se faisant en ajoutant des relations ou des contextes à chaque étape. La différence entre les deux est que dans le Chemin 2 on commence la description des classes par les attributs tandis que dans le Chemin 3 on la commence par les rôles.

Chemin de factorisation. *2*

$0 \Rightarrow [\mathcal{K}_{class}(AOC\text{-}poset)\ \mathcal{K}_{attribute}(AOC\text{-}poset)]$

$1 \Rightarrow [\mathcal{K}_{class}(AOC\text{-}poset)\ \mathcal{K}_{attribute}(AOC\text{-}poset)\ r_{class-attribute}(\exists)]$
$2 \Rightarrow [\mathcal{K}_{class}(AOC\text{-}poset)\ \mathcal{K}_{attribute}(AOC\text{-}poset)\ \mathcal{K}_{role}(AOC\text{-}poset)$
$r_{class-attribute}(\exists)\ r_{role-class}(\exists)]$
$3 \Rightarrow [\mathcal{K}_{class}(AOC\text{-}poset)\ \mathcal{K}_{attribute}(AOC\text{-}poset)\ \mathcal{K}_{role}(AOC\text{-}poset)$
$r_{class-attribute}(\exists)\ r_{role-class}(\exists)\ r_{class-role}(\exists)]$
$4 \Rightarrow [\mathcal{K}_{class}(AOC\text{-}poset)\ \mathcal{K}_{attribute}(AOC\text{-}poset)\ \mathcal{K}_{role}(AOC\text{-}poset)\ \mathcal{K}_{association}(AOC\text{-}poset)$
$r_{class-attribute}(\exists)\ r_{role-class}(\exists)\ r_{class-role}(\exists)]$
$5 \Rightarrow [\mathcal{K}_{class}(AOC\text{-}poset)\ \mathcal{K}_{attribute}(AOC\text{-}poset)\ \mathcal{K}_{role}(AOC\text{-}poset)\ \mathcal{K}_{association}(AOC\text{-}poset)$
$r_{class-attribute}(\exists)\ r_{role-class}(\exists)\ r_{class-role}(\exists)\ r_{association-role}(\exists)]$

Chemin de factorisation. *3*

$0 \Rightarrow [\mathcal{K}_{class}(AOC\text{-}poset)\ \mathcal{K}_{role}(AOC\text{-}poset)]$
$1 \Rightarrow [\mathcal{K}_{class}(AOC\text{-}poset)\ \mathcal{K}_{role}(AOC\text{-}poset)\ r_{class-role}(\exists)]$
$2 \Rightarrow [\mathcal{K}_{class}(AOC\text{-}poset)\ \mathcal{K}_{role}(AOC\text{-}poset)\ \mathcal{K}_{attribute}(AOC\text{-}poset)\ r_{class-role}(\exists)]$
$3 \Rightarrow [\mathcal{K}_{class}(AOC\text{-}poset)\ \mathcal{K}_{role}(AOC\text{-}poset)\ \mathcal{K}_{attribute}(AOC\text{-}poset)$
$r_{class-role}(\exists)\ r_{class-attribute}(\exists)\ r_{role-class}(\exists)]$
$4 \Rightarrow [\mathcal{K}_{class}(AOC\text{-}poset)\ \mathcal{K}_{role}(AOC\text{-}poset)\ \mathcal{K}_{attribute}(AOC\text{-}poset)\ \mathcal{K}_{association}(AOC\text{-}poset)$
$r_{class-role}(\exists)\ r_{class-attribute}(\exists)\ r_{role-class}(\exists)]$
$5 \Rightarrow [\mathcal{K}_{class}(AOC\text{-}poset)\ \mathcal{K}_{role}(AOC\text{-}poset)\ \mathcal{K}_{attribute}(AOC\text{-}poset)\ \mathcal{K}_{association}(AOC\text{-}poset)$
$r_{class-role}(\exists)\ r_{class-attribute}(\exists)\ r_{role-class}(\exists)\ r_{association-role}(\exists)]$

4.3. *Étude qualitative sur le modèle Station Métrologique*

Afin d'évaluer et d'appréhender le comportement de l'ARC pour les quatre chemins décrits précédemment, une première expérimentation a été effectuée sur *Station Métrologique* (cf. Figure 4), sous-modèle décrivant les stations de mesures dans le modèle SIE Pesticides.

Analyse du Chemin Auto Ce chemin conduit à une factorisation maximale en six étapes. Le modèle obtenu correspond au modèle de la Figure 5. Le processus de factorisation de l'ARC a conduit à l'émergence des classes *MeasuringDevice*, *QualityInformation* et *Data* qui factorisent respectivement les attributs *DeviceType* et *DeviceNumber* pour la première, *CodeQuality* pour la seconde et *MeasuringDate* pour la troisième. En outre, l'ARC a permis la factorisation des associations *River Gauging*, *Groundwater Instrumentation* et *Rainfall Instrumentation* de la Figure 4 en une "super-association" *Instrumentation* (cf. Figure 5) entre les classes *MeasuringStation* et la super-classe *MeasuringDevice*. De même, *Water Level Information*, *Rainfall Information* et *Groundwater Information* sont abstraites par la "super-association" *Information* reliant la classe *MeasuringStation* et la super-classe *Data*. Du point de vue métier, ces deux super-associations ont un sens et remplacent dans le modèle final les associations dont elles sont issues. L'ARC produit également une super-association *Monitoring* entre les super-classes *MesuringDevice* et *Data* qui factorise les associations entre les classes (*Water Level Monitoring*, *Groundwater Monitoring* et *Rainfall*

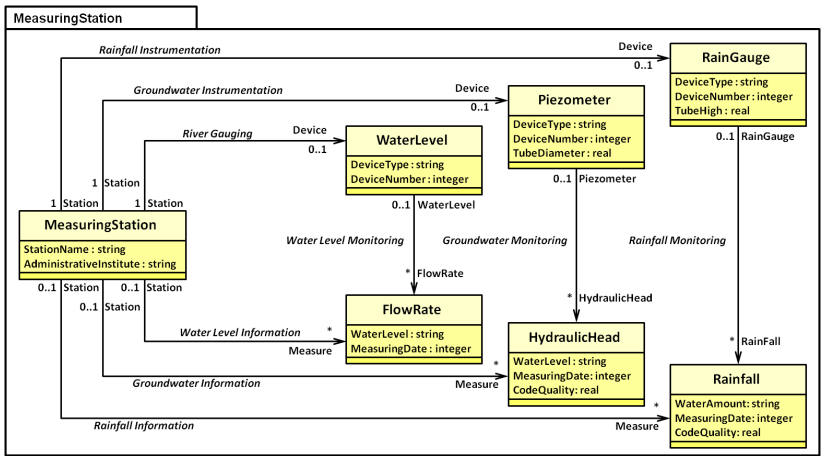


Figure 4. *Sous-modèle* Station Métrologique *(extrait du modèle SIE Pesticides)*

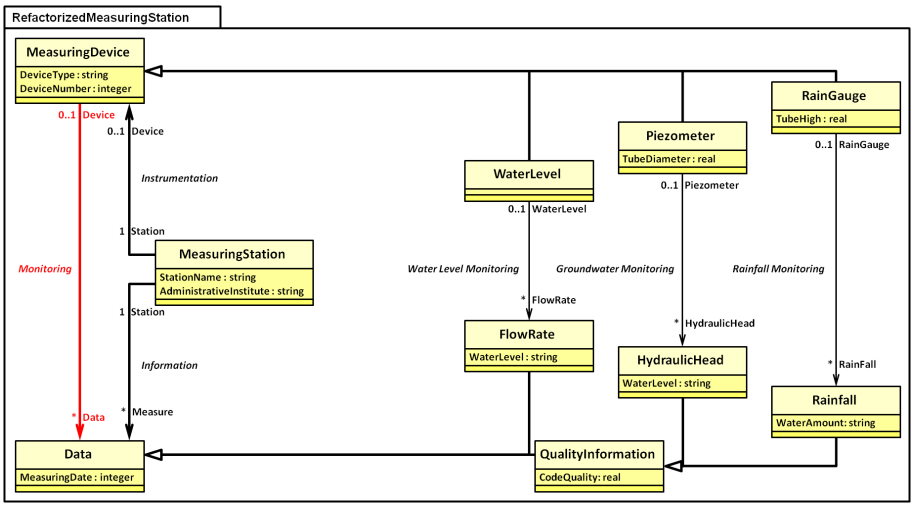


Figure 5. *Sous-modèle factorisé* Station Métrologique

Monitoring) décrivant les instruments de mesure et les données produites par ces instruments. Contrairement aux deux "super-associations" précédentes, cette dernière n'a pas un sens métier assez précis puisque, si elle était adoptée en remplacement des trois autres, elle permettrait à un informaticien peu familier du domaine d'associer le niveau du cours d'eau suivi à un pluviomètre. Par contre, elle est intéressante pour généraliser les associations de monitoring existantes et donner une vue plus générale du modèle. Les trois sous-associations doivent cependant rester pour la préciser (comme UML le permet).

Analyse du Chemin 1 À l'***étape 0***, les relations objet-objet ne peuvent pas être choisies car les treillis n'ont pas été construits. À ce stade, les classes sont à plat dans le treillis de classes et les attributs sont regroupés par leurs noms dans le treillis associé. À l'***étape 1***, les attributs de même nom sont groupés dans des concepts d'abstraction supérieure qui préfigurent en UML les super-classes *MeasuringDevice* et *Data*. À l'***étape 2***, les classes repartent à plat puisque le treillis de classes de l'étape précédente n'est pas pris en compte pour la construction de ce treillis. Les rôles sont quant à eux groupés d'après leurs noms mais aussi d'après le treillis des classes de l'étape d'avant. De ce fait, les rôles *Device* sont factorisés dans une abstraction qui préfigure la super-classe *MeasuringDevice* identifiée à l'étape précédente. De façon analogue, les rôles *Measure* sont factorisés dans une abstraction représentant *Data*. À l'***étape 3***, le treillis de classes à plat de l'étape précédente est ici hiérarchiquement réorganisé par le treillis des rôles précédent. Par exemple, les classes *WaterLevel*, *Piezometer* et *RainGauge* sont ainsi factorisées dans une abstraction prélude du *MeasuringDevice*. À l'***étape 4***, les associations navigables vers des instruments de mesure sont factorisées par le fait que les rôles sont nommés *Device*. Il en est de même pour les associations vers les données puisque le nom de leurs rôles est *Measure*. Contrairement au Chemin Auto, la courbe d'évolution du processus de "factorisation" du Chemin 1 n'est pas asymptotique. Cela s'explique par le fait que, pour une étape donnée, les treillis de l'étape précédente ne sont pas toujours pris en compte dans le calcul. Aussi, la factorisation maximale ne sera donc jamais atteinte.

Analyse du Chemin 2 À l'***étape 0***, comme la relation classe-attribut n'est pas prise en compte dans le calcul, le treillis des classes est plat et celui des attributs aussi. Les attributs de même nom *DeviceNumber*, *DeviceType*, *CodeQuality* et *MeasuringDate* sont factorisés au sein d'abstractions. À l'***étape 1***, le treillis des attributs factorisant les attributs de même nom affecte le treillis des classes où sont matérialisées les super-classes (*MeasuringDevice*, *QualityInformation* et *Data*) regroupant ces attributs. À l'***étape 2***, l'enrichissement du chemin avec la caractéristique rôle n'a aucune influence sur les treillis des classes et des attributs. Celui des rôles laisse entrevoir les futures factorisations issues des rôles de même nom (*Device* et *Measure* en particulier). À l'***étape 3***, l'introduction des relations classe-rôle et rôle-classe n'a pas d'effet majeur en matière de factorisation des classes. À l'***étape 4***, l'ajout au chemin de la caractéristique association n'a aucune influence en matière de factorisation supplémentaire sur les treillis de classes, d'attributs et de rôles. On constate que le treillis des associations est plat. À l'***étape 4***, la nouvelle relation association-rôle du chemin provoque la factorisation des associations au sein du treillis des rôles mais n'a aucun effet sur les trois autres treillis.

Analyse du Chemin 3 À l'***étape 0***, comme aucune relation classe-rôle n'est définie, le treillis des classes est plat, tout comme celui des rôles. Toutefois, les rôles *Device* sont réunis au sein d'un même contexte relationnel tout comme les rôles *Measure*. À l'***étape 1***, le treillis des classes est juste enrichi par les concepts de l'étape 0 factorisant les rôles. À l'***étape 2***, l'introduction des attributs comme caractéristiques ne provoque aucun changement dans les treillis des classes et des rôles. Dans le treillis des attributs, il est possible de constater les regroupements d'une part de *DeviceNum-*

ber et de *DeviceType* qui préfigurent l'abstraction *MeasuringDevice* et, d'autre part, de *CodeQuality* qui donnera la classe *QualityInformation*. La factorisation de l'attribut *MeasuringDate* générera la super-classe *Data*. À l'***étape 3***, les abstractions qui donneront en UML les classes *MeasuringDevice*, *QualityInformation* et *Data* sont matérialisées dans le treillis des classes. À l'***étape 4***, les associations sont introduites mais aucune nouvelle factorisation n'est observée. Le treillis des associations est plat. À l'***étape 5***, les abstractions qui représenteront les associations *Instrumentation*, *Information* et *Monitoring* apparaissent, chacune d'elles factorisant trois associations. Au-delà de cette étape, le processus est effectué automatiquement et les treillis ne sont plus analysés. Comme le processus prend en compte tous les treillis de l'étape précédente, son évolution est semblable au Chemin Auto mais avec un certain retard.

4.4. *Étude quantitative sur le modèle SIE Pesticides*

Dans cette section, nous appliquons l'approche sur le modèle complet et nous mesurons la quantité d'information à traiter par les experts à chaque étape. Nous souhaitons évaluer si la méthode est devenue faisable en pratique comparativement aux approches actuelles en une étape. La version 14 du modèle SIE Pesticides que nous étudions ici est composée de 171 classes ayant au total 130 attributs et 83 associations dont 78 rôles dans le sens de la navigabilité sont nommés et 5 n'ont pas de noms. C'est un modèle dont la connaissance métier du concepteur et la maîtrise du langage UML par les scientifiques impliqués dans le projet ont permis d'organiser sous forme de hiérarchie de nombreuses classes, limitant ainsi le nombre d'attributs par classes mais aussi le nombre d'associations. Ces modèles d'analyse sont créés avec la terminologie des scientifiques et les modèles d'implémentation dérivent de ces derniers par application systématique de transformations dont l'une est renommage des concepts.

Outre le chemin d'exploration automatique, nous avons appliqué à ce modèle les trois chemins décrits ci-dessus. Jusqu'aux étapes 4, pour le Chemin 1, et 5, pour les chemins 2 et 3, les caractéristiques d'entrée de l'algorithme sont celles décrites ci-dessus. Pour les chemins de factorisation 2 et 3, le processus est ensuite effectué automatiquement. La Figure 6 représente, pour chacun des quatre chemins, les évolutions du nombre total des concepts existants et nouveaux à chaque étape.

Nous observons que la courbe du chemin automatique est systématiquement située au-dessus des autres, surtout au cours des premières étapes. Comme tout processus de calcul itératif, son évolution est asymptotique vers un maximum qui sera atteint lorsque la factorisation maximale sera atteinte. Pour le Chemin 1, après une augmentation du nombre total de concepts à l'étape 1, cet indicateur décroît fortement. Cela est dû au fait que, à chaque étape, les caractéristiques d'entrée de l'ARC sont redéfinies sans prendre en compte le niveau de factorisation de l'étape précédente. Cette courbe montre que le processus n'est pas asymptotique et, de ce fait, il est impossible d'obtenir la factorisation maximale des concepts. C'est la raison pour laquelle le processus automatique n'a pas été lancé. Ce chemin a été abandonné pour le reste de l'analyse. Suite à ce constat, les chemins 2 et 3 ont été définis pour que le processus itératif soit

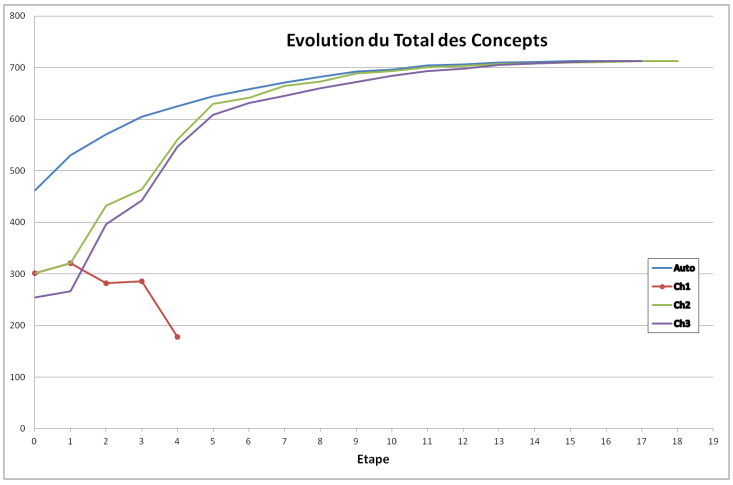


Figure 6. *Évolution du nombre total des Concepts à chaque étape*

cumulatif. Les caractéristiques et les treillis de l'étape n-1 sont pris en compte pour la factorisation de l'étape n. De ce fait, tout comme le chemin automatique, leurs courbes ont une évolution asymptotique et le nombre total de concepts (existants+nouveaux) final est le même pour ces trois chemins (cf. Tableau 1).

Step	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Auto	462	530	571	605	625	644	658	671	682	692	696	704	706	710	711	713	713		
Ch 2	301	321	432	464	560	630	642	665	673	689	693	701	703	707	708	710	711	713	713
Ch 3	254	266	396	442	547	608	631	645	660	672	684	693	698	705	708	711	713	713	

Tableau 1. *Évolution du nombre total de Concepts à chaque étape*

Les chemins 2 et 3 impliquant un nombre de concepts inférieurs à l'étape 0, ils ont un potentiel de factorisation moindre au cours des premières étapes. Toutefois, ces courbes montrent un saut assez marqué entre les étapes 1 et 2. La courbe du Chemin 3 est systématiquement au-dessous des autres. Pour autant, il nécessite une étape de moins que le Chemin 2 (cf. Tableau 1). Étant donné que l'objectif recherché est de limiter le nombre de nouveaux concepts à chaque étape, les Tableau 2 et Tableau 3 montrent cette évolution pour le nombre total des concepts et nombre des classes.

Transition	0->1	1->2	2->3	3->4	4->5	5->6	6->7	7->8	8->9	9->10	10->11	11->12	12->13	13->14	14->15	15->16	16->17	17->18
Auto	68	41	34	20	19	14	13	11	10	4	8	2	4	1	2	0		
Ch 2	20	111	32	96	70	12	23	8	16	4	8	2	4	1	2	1	2	0
Ch 3	12	130	46	105	61	23	14	15	12	12	9	5	7	3	3	2	0	

Tableau 2. *Évolution du nombre total de Concepts*

Le plus grand nombre total de concepts produits (130) est obtenu entre les étapes 1 et 2 du Chemin 3 (cf. Tableau 2), quasiment le double du maximum du chemin

Transition	0->1	1->2	2->3	3->4	4->5	5->6	6->7	7->8	8->9	9->10	10->11	11->12	12->13	13->14	14->15	15->16	16->17	17->18
Auto	32	13	12	6	7	4	5	3	5	0	4	0	2	0	1	0		
Ch 2	20	0	32	0	15	0	11	0	8	0	4	0	2	0	1	0	1	0
Ch 3	12	0	20	18	7	9	4	5	4	4	4	1	3	1	1	1	0	

Tableau 3. *Évolution du nombre de Classes*

automatique (68). Ceci s'explique par le fait que, le nombre maximal de concepts à atteindre (713) est constant et ce quel que soit le chemin. En effet, les chemins 2 et 3 démarrant avec un nombre de concepts inférieurs, le processus "rattrape" son retard de façon plus "brutale". Toutefois, le nombre de nouvelles classes entre deux étapes est mieux réparti et inférieur pour le Chemin 3 que pour les autres chemins (cf. Tableau 3). Pour les chemins Auto et 2, le plus grand nombre de nouvelles classes est 32 alors que, pour le Chemin 3, il est 20. L'inconvénient est, qu'après l'étape 5, il reste encore 37 classes à découvrir pour le Chemin 3 et seulement 24 et 27 respectivement pour les chemins Auto et 2. Une analyse métier approfondie des classes restant à découvrir après l'étape 5 pour les chemins 2 et 3 doit être menée avec les scientifiques impliqués dans le projet SIE Pesticides afin d'évaluer leur intérêt métier.

5. Travaux connexes

L'élimination des doublons est un sujet qui revêt différentes formes dans le domaine de la modélisation comme dans celui de la programmation. C'est notamment l'une des opérations de *refactoring* les plus connues (Fowler, 1999 ; Opdyke et Jonhson, 1993) et un cas particulier de la notion de clones dans le code (Roy *et al.*, 2009). Elle est au cœur d'approches visant à la reconstruction de hiérarchies de classes, de manière globale (Casais, 1991 ; Cook, 1992 ; Cherfi et Lammari, 2002) comme de manière incrémentale (Bergstein et Lieberherr, 1991). Elle s'accompagne de l'extraction d'abstractions organisées dans une hiérarchie de spécialisation, comme le développent certaines approches de rétro-ingénierie de modèles de bases de données relationnelles vers des modèles conceptuels (Akoka *et al.*, 1999).

L'Analyse Formelle de Concepts et les treillis en général ont été largement utilisés dans ce même contexte. À notre connaissance, les premiers usages des treillis datent de travaux dans le domaine de la refactorisation des schémas de bases de données orientés objets (Missikoff et Scholl, 1989 ; Rundensteiner, 1992). Dans le domaine de la programmation, l'AFC a été mise en œuvre pour extraire des interfaces abstraites à partir d'une hiérarchie de classes Smalltalk (Godin et Mili, 1993) ou pour refactoriser une hiérarchie de classes (Dicky *et al.*, 1996). Nous avons présenté dans la section 2 plusieurs approches utilisant l'ARC dans ce même domaine. Cet article poursuit dans cette voie, mais en tâchant d'améliorer sa faisabilité pratique et en travaillant par étapes plutôt que par analyse d'un résultat global.

6. Conclusion

Dans cet article, nous proposons une nouvelle approche afin de contrôler le processus de factorisation de certaines caractéristiques (classe, attribut, rôle et association) au sein d'un modèle de classes. Pour des modèles de taille conséquente, l'utilisation classique de l'Analyse Relationnelle de Concepts produit souvent un nombre important de nouveaux concepts qui rend difficile leur analyse métier par un expert du domaine. Nous simplifions cette analyse en limitant le nombre de concepts apparaissant à chaque étape du processus itératif par un choix de chemin de factorisation. Les résultats de factorisation sont comparés au processus classique. Les premiers résultats montrent que les chemins ne peuvent pas être choisis au hasard et que, si on souhaite atteindre une factorisation maximale sans diverger, il faut ajouter les caractéristiques de façon cumulative. L'objectif de réduire le nombre de concepts par étape est atteint dans les premières étapes contrôlées manuellement mais, pour atteindre la factorisation maximale, le processus va soit procéder par un saut à une étape ultérieure soit répartir le différentiel tout le long du chemin. Dans ce second cas, il est plus facile à un expert d'analyser les nouveaux concepts issus de la factorisation. Au stade actuel, il est impossible de prévoir l'un ou l'autre de ces deux comportements. Une expérimentation plus systématique sera menée pour cela.

Finalement, il faut noter que l'ordre d'introduction des caractéristiques modifie l'ordre des factorisations et que la reconstruction du modèle UML à chaque étape pourrait accélérer la convergence vers la factorisation maximale d'autant plus que certaines factorisations ne sont pas intéressantes du point de vue métier. La méthode d'analyse ARC est très sensible à l'orthographe, à la sémantique (polysémie, synonymie, parasynonymie...), etc. Cela constitue une limite de l'approche actuelle. Aussi, nous envisageons dans une prochaine étape d'enrichir les données d'entrée de l'ARC et de "piloter" ces méthodes par des ontologies métiers ou des techniques de Traitement Automatique du Langage Naturel. L'introduction des ontologies facilitera en particulier le nommage des nouveaux concepts générés et évitera en partie une expertise humaine.

Remerciements

Ce travail a été financé par l'ANR11_MONU14 Fresqueau et le projet Miriphyque du programme Pesticides du MEEDDM.

7. Bibliographie

Akoka J., Comyn-Wattiau I., Lammari N., « Relational Database Reverse Engineering : Elicitation of Generalization Hierarchies », *ER (Workshops)*, 1999, p. 173-185.

Bergstein P., Lieberherr K., « Incremental Class Dictionnary Learning and Optimization », *ECOOP'91*, 1991, p. 371–396.

Casais E., « Managing Evolution in Object Oriented Environments : An Algorithmic Approach », Thèse de doctorat, Université de Genève, 1991.

Cherfi S. S.-S., Lammari N., « Towards and Assisted Reorganization of Is-A Hierarchies », *Object-Oriented Inf. Systems*, vol. 2425 de *LNCS*, Springer-Verlag, 2002, p. 536–548.

Cook W., « Interfaces and Specifications for the Smalltalk-80 Collection Classes », *OOPSLA'92*, 1992, p. 1–15.

Dicky H., Dony C., Huchard M., Libourel T., « On Automatic Class Insertion with Overloading », *OOPSLA 96*, 1996, p. 251–267.

Dolques X., Ber F. L., Huchard M., Nebut C., « Analyse Relationnelle de Concepts pour l'exploration de données relationnelles », *EGC*, 2013, p. 121-132.

Falleri J.-R., Huchard M., Nebut C., « A Generic Approach for Class Model Normalization », *ASE 2008*, 2008, p. 431-434.

Fowler M., *Refactoring : Improving the Design of Existing Code*, Add.-Wesley Prof., 1999.

Ganter B., Wille R., *Formal Concept Analysis : Mathematical Foundation*, Springer-Verlag Berlin, 1999.

Godin R., Mili H., « Building and Maintaining Analysis-Level Class Hierarchies Using Galois Lattices », *OOPSLA 93*, 1993, p. 394–410.

Guédi A. O., Miralles A., Huchard M., Nebut C., « A Practical Application of Relational Concept Analysis to Class Model Factorization : Lessons Learned from a Thematic Information System », *Concept Lattices and Their Applications (CLA 2013)*, 2013, p. 9-20.

Hacène M. R., Huchard M., Napoli A., Valtchev P., « Relational concept analysis : mining concept lattices from multi-relational data », *Ann. Math. Artif. Intell.*, vol. 67, n° 1, 2013, p. 81-108.

Hacène M. R., « Relational concept analysis, application to software re-engineering », Thèse de Doctorat, Université du Québec À Montreal, 2005.

Miralles A., Pinet F., Carluer N., Vernier F., Bimonte S., Lauvernet C., Gouy V., « EIS-Pesticide : an information system for data and knowledge capitalization and analysis », *Euraqua-PEER Scientific Conference, 26/10/2011 - 28/10/2011*, Montpellier, FRA, 2011.

Missikoff M., Scholl M., « An Algorithm for Insertion into a Lattice : Application to Type Classification », *Proceedings of the 3rd Int. Conf. FODD'89*, , 1989, p. 64–82.

Opdyke W., Jonhson R., « Creating Abstract Superclasses by Refactoring », *Proc. of the 21st Annual Conference on Computer Science, Indianapolis (IN), USA*, ACM Press, New York (NY), USA, 1993, p. 66–72.

Petersen W., « A Set-Theoretical Approach for the Induction of Inheritance Hierarchies », *Proc. of FG/MOL-01), ENTCS*, vol. 53, Elsevier, July 2001, p. 296-308.

Roume C., « Analyse et restructuration de hiérarchies de classes », Thèse de Doctorat, Université Montpellier 2, 2004.

Roy C. K., Cordy J. R., Koschke R., « Comparison and evaluation of code clone detection techniques and tools : A qualitative approach », *Sci. Comp.. Prog.*, vol. 74, n° 7, 2009, p. 470-495.

Rundensteiner E. A., « A Class Classification Algorithm For Supporting Consistent Object Views », rapport, 1992, University of Michigan.

Vernier F., Miralles A., Pinet F., Carluer N., Gouy V., Molla G., vin Petit K., « EIS Pesticides : An environmental information system to characterize agricultural activities and calculate agro-environmental indicators at embedded watershed scales », *Agricultural Systems*, vol. 122, 2013, p. 11-21.

Atlas géomatique collaboratif pour l'environnement et la gestion durable des ressources halieutiques, en Afrique de l'ouest, cas de la Mauritanie

Elaboration d'un système d'information collaboratif

Ely Beibou[1], Jérôme Guitton[2], Thérèse Libourel[3]

1. Institut Mauritanien de recherches Océanographiques et des Pêches (IMROP)
BP 22, Nouadhibou, Mauritanie
beibou_es@yahoo.fr

2. UMR ESE, Ecologie et Santé des Ecosystèmes
Agrocampus ouest, centre de Rennes, France
jerome.guitton@agrocampus-ouest.fr

2. UMR 228 ESPACE-DEV (IRD - UM2)
500 rue Jean-François Breton-34093 Montpellier Cedex 5
therese.libourel@univ-montpellier2

RÉSUMÉ. L'intérêt de la géomatique collaborative pour la gestion de l'environnement et des ressources halieutique est capital. Or, dans la zone ouest-africaine et en particulier, en Mauritanie, les informations spatiales relèvent des systèmes d'information disparates et ne peuvent pas être facilement mises à profit dans les politiques publiques de gestion de l'environnement et des ressources renouvelables. C'est dans ce contexte que s'inscrit ce travail, visant à concevoir un système d'information collaboratif orienté pêche et environnement. Après l'introduction du contexte général, nous présentons la problématique de recherche, l'état de l'art sur les approches collaboratives existantes et nous relatons les étapes de la construction d'un atlas et les résultats obtenus avant de conclure.

ABSTRACT. The interest of collaborative geomatics for management of the environment and fisheries resources is crucial. However, in the West African region and in particular Mauritania, spatial information systems fall of disparate information systems and cannot be easily put to use in public policy management of environment and renewable resources. It's in this context that fits this work, which aims to develop a collaborative information system oriented fisheries and environment. After the introduction of the general context, we present the research problem, the state of the art on existing collaborative approaches and we report on the steps of the construction of an atlas and present the results before concluding.

MOTS-CLÉS : Système d'information et de connaissance – Géomatique collaborative – Environnement - Ressources halieutiques – Mauritanie - Afrique de l'Ouest.

KEYWORDS: Information and knowledge System – Collaborative Geomatics – Environment – Fisheries resources - Mauritania - West Africa.

1. Introduction

Lors du sommet de Rio de Janeiro 1992, pour aborder les problèmes qui préoccupent la communauté mondiale (détérioration de l'environnement, déforestation, pollution, épuisement des stocks de poissons, etc.), les participants ont souligné l'importance et la nécessité de l'utilisation de l'information spatialisée. Suite à cela, le sommet mondial sur le développement durable tenu à Johannesburg en 2003, a été l'occasion de souligner et d'illustrer, au travers de cas pratiques, les capacités et les potentialités de l'utilisation de l'information géographique en appui à la gestion de l'environnement et au développement durable.

L'utilisation des données géographiques pour la préservation de l'environnement et l'aménagement durable des pêcheries, revêt une importance capitale pour les pays de la Commission Sous Régionale des Pêches (CSRP[1]), qui sont des pays côtiers dont les eaux maritimes sont convoitées par différentes flottilles (industrielles, artisanales, côtières, nationales et étrangères). Dans ce contexte, l'intérêt pour ces pays, à renforcer leur capacité à accéder, intégrer et utiliser les données géographiques provenant de sources diverses, pour guider la prise de décision à toutes les échelles, devient crucial. D'autant plus que leur capacité à prendre collectivement des décisions éclairées à tous les niveaux, est tributaire des possibilités d'accès aux données spatiales et aux services (traitements) de celles-ci.

La gestion des pêches en Afrique de l'Ouest constitue, à nos yeux, un exemple type. Celle-ci s'avère particulièrement complexe car, les échelles des stocks de ressources halieutiques sont diversifiées, les acteurs sont multiples et de ce fait, son périmètre dépasse celui de la gestion de l'information halieutique, *stricto sensu*. Ainsi, souvent les stocks sont partagés entre plusieurs pays, alors que la gestion de l'information halieutique se passe, dans le meilleur des cas, à un niveau national. De la même manière, les thématiques utiles à la gestion des stocks sont portées par des organismes divers (scientifiques, entreprises, comités de pêcheurs, gouvernement, états tiers ayant des autorisations de pêche...).

En ce qui nous concerne, en Mauritanie plus spécifiquement, le département des pêches mène une réflexion qui vise à travers l'utilisation de ces données, à renforcer sa capacité à assurer une gestion durable de ses ressources halieutiques et à préserver l'environnement marin et côtier dans une approche écosystémique. Cette stratégie cherche à pallier l'état persistant de surexploitation des principaux stocks

[1] CSRP : La Commission Sous-Régionale des Pêches est un organisme intergouvernemental. Elle regroupe le Cap Vert, la Gambie, la Guinée, la Guinée Bissau, la Mauritanie, le Sénégal et la Sierra Leone.

de poissons et à empêcher l'évènement d'une éventuelle dégradation de l'environnement, dans un contexte marqué par l'intensification et la diversification des activités extractives (pêche, exploration et exploitation pétrolières offshore).

Dans ce cadre, notre proposition s'appuie sur la conception et la mise en œuvre d'une plateforme collaborative et spatiale permettant l'amélioration des échanges entre les différents acteurs et intervenants et donc la construction d'un diagnostic partagé, préalable à la mise en place de plans de gestion. Cette initiative devrait avoir un impact positif sur la gestion des ressources halieutiques dans cette zone du monde.

Dans cet article, nous présentons cette initiative. Dans un premier temps, au niveau de la section 2, nous préciserons le contexte, les objectifs et la problématique de recherche. La section 3 présentera un état de l'art dédié aux initiatives menées plus spécifiquement en géomatique, la dimension spatiale étant primordiale. La section 4 sera dédiée à notre proposition, aux spécifications et réalisation de l'application ***atlas géomatique collaboratif*** et la section 5 conclura notre propos.

2. Problématique de recherche

Notre défi est de mettre à profit toutes les possibilités offertes par l'ensemble des systèmes d'information existants dans la région, et en particulier les systèmes d'information géographiques, pour appuyer le processus de décision.

Plusieurs systèmes d'information des pêches ont été mis en place dans la sous région. Nous allons nous focaliser sur le cas de la Mauritanie. Notre objectif à long terme est d'intégrer ces systèmes d'information par interaction **contrôlée** de services divers. La spatialisation de l'information et son rendu cartographique étant reconnus comme moyens de communication privilégiée entre acteurs, cela nous a amené à un premier sous objectif à court terme consistant à mettre en place une application dénommée *atlas en ligne* qui met en œuvre la collaboration des composants du système global et de ses acteurs.

Etant dans le domaine de la recherche en systèmes d'information, notre problématique générale est celle de l'intégration d'information et de connaissances.

Nous pouvons résumer les verrous principaux comme suit :

- la divergence voir le caractère conflictuel des objectifs assignés à chacun de ces systèmes d'information (publics : aménagement et développement du secteur, privés : de recherche de bénéfices, société civile : protection de l'environnement, etc.) ;
- la dispersion spatiale : répartition géographique des acteurs propriétaires des systèmes d'information en question ;
- l'hétérogénéité : la spécificité, tant au niveau technique (par exemple, la plate-forme matérielle, système d'exploitation, etc.) qu'au niveau conceptuel (par exemple, modèle de données, langage de requête, etc.) ;

- autonomie : ces systèmes d'information sont autosuffisants, et chaque acteur préfère garder ses particularités au lieu de jouer un rôle uniquement en tant que composant dans un système plus vaste.

Pour aborder la question de l'interopérabilité recherchée, deux voies de solution se présentent à nous : l'entreposage et la médiation (figure 1).

Dans le cas de l'entreposage (Rundensteiner, Koeller, et Zhang 2000), les données sont extraites, mises en forme et stockées dans un entrepôt centralisé. Ensuite, les requêtes sont adressées à cette nouvelle base de données et le système entrepôt est déconnecté des sources initiales (du moins entre deux mises à jour). Cette approche a le bénéfice d'être performante en termes de rapidité d'exécution des interrogations et des traitements. Par contre, la mise à jour des sources demande un rechargement régulier de l'entrepôt (ce qui peut être coûteux) et peut même nécessiter une évolution du schéma de l'entrepôt. Les résultats des traitements sont donc dépendants de l'état de l'entrepôt.

La seconde voie est celle de la médiation (Wiederhold 1992). Les données sont extraites directement par requêtes à partir des sources de données et les traitements leur sont appliqués en temps réel, ce qui fait que les résultats sont mis à jour dynamiquement en fonction de l'état des sources. Par contre, la performance globale du système, dans ce cas, peut chuter car les sources sont sollicitées à chaque requête.

Notre proposition se situe sur une ligne intermédiaire entre les deux approches. Une idée globale du système envisagé est schématisée en figure 1. Les sources de données diverses et disparates restent autonomes. La médiation se fait par l'intermédiaire de services. Dans cet article, nous nous focaliserons sur la conception et la mise en œuvre du service dénommé atlas en ligne en nous appuyant sur l'innovation issue de la géomatique collaborative.

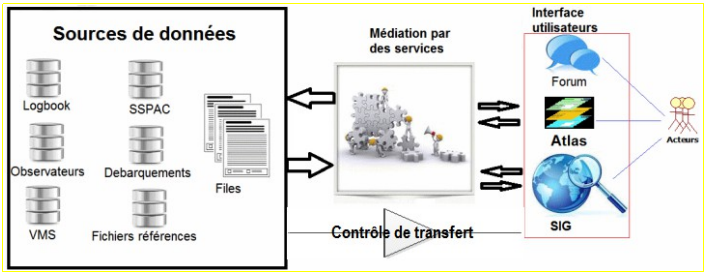


Figure 1 : vision globale du système pêches en Mauritanie

3. Etat de l'art

Notre objectif étant celui d'une application collaborative à dimension spatiale, nous nous sommes intéressés à tous les travaux existant autour de la cartographie sur le web et à leurs avancées.

La création participative des cartes a commencé en fin des années 1980. A cette époque, la priorité était de susciter les connaissances autochtones et locales en utilisant la dynamique communautaire afin de faciliter la communication entre les populations locales et les chercheurs. Dès l'aube des années 1990, un nouvel environnement, marqué par le développement des SIG[2] et des nouvelles technologies de l'information et de la communication, a facilité l'émergence des initiatives communautaires à base des outils géo-informatiques. Cette situation a favorisé l'apprentissage social des techniques géo-informatiques, la communication bidirectionnelle (société vs chercheurs), l'élargissement de la participation du public à travers ce qu'on appelle (SIGP : SIG participatifs). (Giacomo Rambaldi et al., 2004, source : http://www.iapad.org/participatory_gis.htm).

Les PPGIS verront le jour, dans le prolongement du courant des critiques sur les SIG, en plein milieu des années 1990, avec le lancement par l'ANCGIA (American National Center for Geographic Information and Analysis) de sa 17 ème initiative portant sur la prise de décision spatiale collaborative. L'objectif était alors, d'étendre les cadres conceptuels des systèmes spatiaux d'aide à la décision (SDSS) pour supporter des groupes de décideurs dans la recherche de solutions propices aux problèmes spatiaux complexes (Sieber 2006). Une nouvelle notion de CSCW (Computer-Supported Cooperative Work) est introduite par (Robert Laurini 1998) et un SIG Collaboratif par visioconférence pour la gestion de l'espace par (Cowen et al., 1998). Ensuite, émerge la notion de TeleGeoProcessing, issue du SIG et des Télécommunications, considérée par (R. Laurini, Servigne, et Tanzi 2001) comme une nouvelle discipline caractérisée par les bases de données spatiales, la cartographie à la demande, l'échange d'information entre différents sites à l'aide de tout type et système de communication et d'analyse de données spatiales en ligne. Ils (R. Laurini, Servigne, et Tanzi 2001) considèrent le TeleGeoMonitoring comme extension du TeleGeoProcessing faisant intervenir l'usage des outils de positionnement (GPS[3]), les bases de données spatiales et groupes de prise de décisions dynamiques. Vient ensuite le projet européen GeoMed, initié par l'Allemagne, la Belgique (Volkmar Pipek et al., 2000) et (Batita et al. 2012). Le modèle ou carte d'argumentation ou délibérative apparaît avec (Rinner 2001) qui, introduisaient ce modèle comme manière de structurer les débats en support au processus de prise de décision. A partir des années 2000, commence l'ère des travaux portant sur la géocollaboration (Batita et al. 2012).

Au vu de cette évolution, nous synthétisons et distinguons, de notre point de vue, les trois étapes importantes de l'évolution de la géomatique collaborative :

La première, que nous qualifierons de **période d'adaptation** à divers profils utilisateurs dans un contexte de système fermé, s'est produite en plein milieu des années 1990. A cette époque, les SIG ont évolué d'une perspective d'expertise purement technique vers des SIG plus participatifs donnant naissance au terme du PPGIS (Public Participatory GIS). Ce terme a vu le jour lors de deux réunions en 1996 du NCGIA (National Center of Geographic Information and Analysis).

[2] Système d'Information Géographique.
[3] Global Positioning System

Constatant le niveau jugé satisfaisant d'appropriation des techniques SIG par des utilisateurs non avertis. Lors de ces réunions, il a été question d'encourager les franges défavorisées des populations à utiliser les SIG dans la perspective de d'améliorer la transparence et d'influencer les politiques gouvernementales (Obermeyer, 1998). On note que pendant cette période, l'approche traditionnelle « top-down » a commencé à céder petit à petit la place à l'approche horizontale [en anglais : bottom-up], ce qui va de pair avec les objectifs participatifs (Sieber 2006) et (Batita et al. 2012).

La seconde période, que nous qualifierons de **période d'appel à contribution**, dans un contexte de système ouvert, permettant l'interaction utilisateur/système, est caractérisée par l'évolution des SIG de l'approche PPGIS vers une approche VGI (Volunteered Geographic Information). Il s'agit de systèmes utilisant le web pour créer, assembler et disséminer l'information géographique provenant volontairement des individus (http://www.ncgia.ucsb.edu/projects/vgi). Cette évolution a concrétisé l'ancrage de l'approche ascendante au sein des SIG. Ainsi, on assiste à une transformation radicale dans le comportement et l'usage que font les citoyens qui sont maintenant capables de produire et introduire, sans assistance, l'information, et d'observer et décrire le monde, à l'aide des nouvelles technologies géoinformatiques. L'exemple du projet OpenStreetMap marque le tournant de l'évolution des outils SIG du participatif encadré vers une approche contributive volontaire ou spontané.

La troisième période, que nous qualifierons de **période d'interaction**, dans un contexte de système ouvert, permettant l'interaction utilisateur/utilisateur, médiée par le système (interactions internes), c'est celle de la fin des années 2000 à nos jours où, les SIG évoluent vers les WikiGIS ou WikiSIG. Ce dernier, conçu pour supporter efficacement le travail collaboratif spatialisé et produire de l'information géographique tout en documentant et visualisant cartographiquement l'ensemble des contributions des acteurs impliqués, s'appuie sur un système de gestion des contenus (CMS) de type wiki (Batita et al. 2012). Le WikiGIS est un SIG, construit en ligne par des interventions collectives, lesquelles supposent des interactions entre les participants, puis la fusion et la traçabilité de leurs contributions dans des représentations géospatiales cohérentes et ouvertes à l'enrichissement. Ces représentations géospatiales constituent l'expression des connaissances collectives sur un territoire ou sur un phénomène spatialisé donné (Ciobanu et al., 2007). Ce qui renvoie au changement du paradigme du contributif volontaire au contributif encadré ou supervisé.

En conclusion, nous nous situons résolument dans la troisième période. Nous nous sommes aussi inspirés de divers travaux menés par des organismes internationaux Ifremer, IRD et UEOMA (cf. section 4).

4. Notre proposition

4.1 Méthodologie

Après avoir établir un état de l'art des travaux existants autour de la cartographie sur le web et de la notion de collaboration, nous avons étudié, analysé et fait un état des lieux de l'existant en termes de données et de systèmes d'information dans la sous région. Cela nous a permis de décrire toutes les sources de données existantes, d'identifier celles qui sont nécessaires à la création d'indicateurs, d'identifier et classifier les utilisateurs potentiels et d'analyser leurs besoins respectifs en matière d'accès aux données, de partage de ressources et enfin de définir leurs rôles dans la collaboration autour de notre problématique.

La phase suivante a consisté à affiner l'architecture de la plateforme pressentie, ainsi que les fonctionnalités adaptées aux besoins identifiés dans la phase précédente.

Ensuite, nous avons réalisé un prototype fonctionnel (dénommé atlas en ligne) dotés des services web nécessaires pour l'accès aux indicateurs (données, graphiques ou couches géographiques) par des clients externes et muni d'outils de collaboration répondants aux besoins de notre problématique (forum de discussion).

Pour évaluer notre prototype nous comptons l'utiliser pour analyser un problème de conflit sur l'espace de deux pêcheries artisanales ciblant le poulpe en Mauritanie, à l'aide de deux engins différents (le pot et la turlutte) et pratiquées par les nationaux (pot à poulpe) et les étrangers (turlutte). L'expérimentation mettra ensemble les acteurs (forum de discussion) pour mesurer le niveau de cohabitation possible et confrontera les données (atlas en ligne) pour estimer les impacts écologiques et socioéconomiques de ces activités, en vue d'éclairer les décideurs.

4.2 Analyse de l'existant

4.2.1 Les données

Au niveau de la Mauritanie, plusieurs organismes dépendants du Ministère des Pêche et de l'Economie Maritime (MPEM) disposent de sources de données disparates. La figure 2 détaille toutes celles-ci ainsi que les organismes concernés.

Il faut noter que le suivi des activités de pêche en Mauritanie est pris en charge de deux manières distinctes. La première, obligatoire pour toute activité de pêche industrielle, est assumée par les Garde-côtes mauritaniens (GCM). Elle comprend les déclarations émanant des patrons pêcheurs (les logbooks), rapportant les actions de pêches et les captures au niveau de chaque rectangle statistiques mauritanien (grille de ½° par ½°) et le suivi satellitaire des navires de pêche (VMS : Vessel Monitoring System).

Le second système de suivi est adossé à la recherche de manière plus spécifique c'est-à-dire que ce sont les besoins de la recherche qui dictent son

dimensionnement et c'est aussi les équipes de l'organisme de recherche (IMROP) qui les mettent en œuvre. Ce deuxième système d'observation comprend principalement le système de suivi de la pêche artisanale et côtière (SSAPC) qui met en œuvre des protocoles d'échantillonnage stratifié des activités de pêche artisanale. Il supporte aussi la collecte des observateurs embarqués à bord des bateaux de pêche industrielle pour récolter les données fines servant à compléter et détailler les données déclaratives (logbook).

4.2.2 Les utilisateurs

Les utilisateurs potentiels de cet outil relèvent de trois (quatre si on rajoute le grand public) catégories.

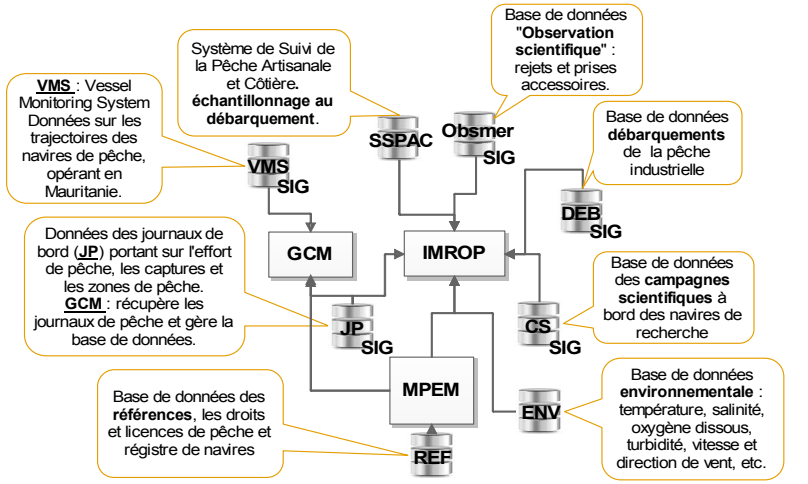
Les gestionnaires. Ce groupe comporte les administrateurs du secteur des pêches poursuivant les objectifs des pouvoirs publics. Ils sont souvent intéressés par des informations synthétiques et des tendances globales sous des formes simples et utilisables sans effort. Ils se contentent d'une vision synthétique et unifiée des problématiques.

Les professionnels. Ce groupe compte les administrateurs privés et leurs collaborateurs (pêcheurs, mareyeurs, etc.) poursuivant des objectifs d'augmentation des bénéfices tirés de leurs activités extractives. Il détient les connaissances et l'expertise acquises au fil des années, à travers la pratique de l'activité sur le terrain. Ce groupe, dans le cadre de la concertation, pourra apporter son expertise au sein des fiches thématiques[4] (nous y reviendrons plus loin). Soit en les considérant comme des rédacteurs des fiches, soit en tant qu'utilisateurs privilégiés du système de commentaires, rendu accessible au sein des fiches et qui permet de capter l'expertise des acteurs.

Les scientifiques. Ce groupe est composé des chercheurs et scientifiques, public ou privés, intéressés par le secteur. Ils ont souvent des compétences avérées dans l'utilisation des technologies. Ce groupe ne se contente pas des informations synthétisées et préfère souvent disposer des données sous différentes formes (y compris celle d'origine) afin de pouvoir potentiellement leur appliquer d'autres traitements pour répondre à leurs besoins spécifiques.

Et bien évidemment le **grand public** qui pourra accéder à l'information.

[4] Celles-ci sont les éléments constitutifs de l'atlas (cf. 4.3)



MPEM : Ministère des Pêches et de l'Economie Maritime
GCM : Gardes Côtes Mauritaniennes (Ex. DSPCM : Délégation à la Surveillance des Pêches et au Contôle en Mer)
IMROP : Institut Mauritanien de Recherches Océnographiques et des Pêches
SIG : Base de données spatiale (géoréférencée)

Figure 2 : principales sources d'information des pêches en Mauritanie

4.3 Architecture

L'architecture (figure 3) de notre proposition repose sur la vision générale présentée en section 2. Elle est composée de 3 niveaux dont le premier (Sources) comporte les différentes sources disparates d'information, le second (Médiation services) est constitué de l'ensemble des services de médiation qui vont assurer l'interface entre ces différentes sources et les applications clientes. Le dernier niveau (Médiation applications) est composé des applications clientes intégrées qui sont i) un atlas en ligne pour la gestion et le suivi des connaissances produites à partir des données et informations stockées dans les différentes sources ; ii) une interface cartographique pour mener à bien l'analyse des informations spatialisées et ; iii) un forum de discussion et d'échange, pour assurer la collaboration entres les acteurs autour des thèmes liés à l'aménagement durable des ressources et à la protection de l'environnement. Sur ce niveau pourront venir se connecter d'autres applications externes à l'aide des services WMS pour accéder aux indicateurs spatialisés et des services web pour accéder aux données de chaque indicateur indépendamment de sa mise en forme.

Ce que nous proposons est donc une architecture innovante qui respecte l'autonomie de chaque source de données tout en assurant l'interopérabilité syntaxique et sémantique des différents composants sans intégration physique au sein d'un système unique.

La médiation se fait, pour l'accès aux données distantes, par l'intermédiaire des services web qui permettent de contrôler les utilisateurs par origine (adresse IP) ou identification (Login / mot de passe). L'intérêt de l'approche réside dans le caractère dynamique de l'application. Les mises à jour à partir des sources originelles sont, en mode « connecté », prises en compte en temps réel. L'autre avantage de la méthode est la possibilité de basculer entre mode « connecté » et « déconnecté ». Pour des contraintes de performance (temps d'accès à la donnée fine), l'application peut aussi passer en mode déconnecté qui consiste à mettre à jour les indicateurs et à garder les résultats en cache (on entrepose les données correspondant à l'indicateur final). Cette solution permet de passer outre les contraintes des architectures matérielles et logicielles des systèmes d'information des acteurs, tout en gardant un niveau de performance acceptable, en terme de temps de réponse aux interrogations invoquées par la consultation des indicateurs.

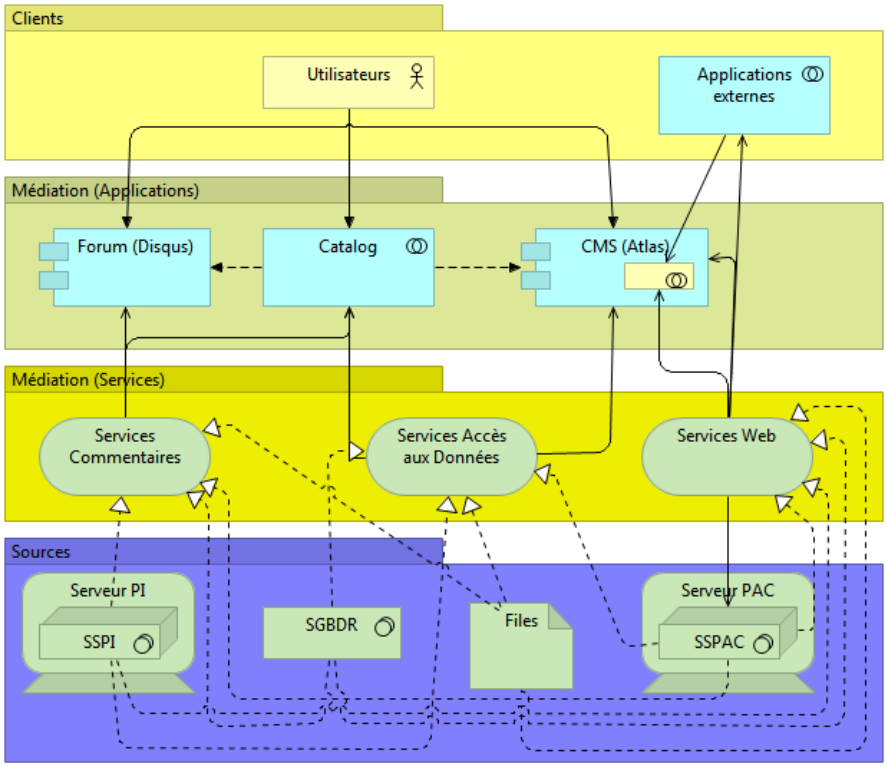


Figure 3 : architecture globale du système

4.4 Première réalisation : atlas en ligne collaboratif

Sur le plan applicatif, nous allons utiliser une application existante développée en PHP, et utilisée déjà dans plusieurs projets (CHARM[5], Atlas thonier[6], atlas UEOMA[7]). L'idée est de compléter cette application en lui ajoutant de nouvelles fonctionnalités de collaboration. Ce prototype fonctionnel est composé d'un module qui permet l'interrogation de bases de données hétérogènes et le stockage des résultats en XML dans un entrepôt de données. L'application permet la réutilisation de ces données pour créer des indicateurs (graphiques) intégrés au sein de fiches thématiques (HTML) organisées et assemblées à l'aide de transformations XSL.

Une partie de la proposition consistera dans le développement d'outils de collaboration. La seconde partie consistera à implémenter des services web d'accès aux indicateurs (données, graphiques ou couches géographiques) de manière indépendante de l'atlas. Enfin, la dernière brique à développer sera composée d'un outil de gestion des métadonnées qui permettra la description des données de base ainsi que des indicateurs produits.

La figure 4, ci-dessous, donne une vision synthétique des diverses étapes permettant l'accès en ligne à une fiche thématique. L'application peut se connecter à une ou plusieurs bases de données pour en extraire, à l'aide de requête prédéfinies adaptées aux besoins identifiés des utilisateurs, les jeux de données nécessaires à la construction d'indicateurs présentés sous différentes formes au sein des fiches thématiques. Nous donnons ensuite un ensemble de détails éclairant le propos.

4.4.1 Construction de fiches thématiques

La mise en forme de la fiche telle qu'elle apparaît dans la figure 4 est définie dans une feuille XSL où, l'on détermine l'ordre d'apparition des indicateurs et où l'on fait appel à la procédure d'affichage des indicateurs (affiche_graphique.php), en passant le numéro de l'indicateur concerné (no_graph) ici égal à 1 i.e qu'il sera le premier indicateur sur la fiche (cf. figure 5).

Chaque indicateur (qui sera présent dans une fiche) est structuré et renseigné selon trois parties principales (au sein du modèle XML générique de fiche) qui contiennent les métadonnées, les paramètres de mise en forme de l'indicateur et les données nécessaires à sa construction.

L'application génère automatiquement les métadonnées des indicateurs au fur et à mesure qu'ils sont créés. Pour chaque indicateur est renseigné son titre ou nom, le type de fiche auquel il est associé, son objectif, sa représentation graphique, un guide de lecture, l'origine de la donnée utilisée, l'échelle géographique, l'échelle temporelle, les données mobilisées, et la requête SQL qui permet l'obtention des données.

[5] http://charm-project.org/en/toolsmenu/fisheries-atlas/fisheries-atlas-tool
[6] http://sirs.agrocampus-ouest.fr/atlas_thoniers/
[7] http://atlas.statpeche-uemoa.org/

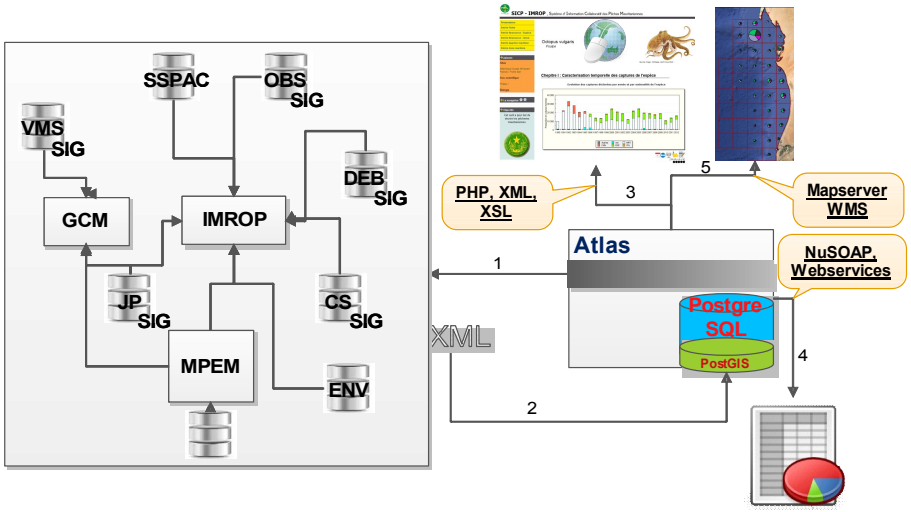


Figure 4 : fonctionnalités de l'atlas 1 – interrogations (en local ou à distance par service web) ; 2 –obtention de résultat en XML ; 3 – construction de fiche à partir des résultats obtenu en 2 ; 4 – envoi des résultats (numériques ou textuels) par service web à d'autres client ; 5 – envoi des cartes par WMS.

```
<div class='cellule'>
<p class='titrecellule'>Evolution des captures déclarées par année et par nationalité de l'espèce</p>
<xsl:call-template name="affiche_graphique">
<xsl:with-param name="no_graph" select="1" />
</xsl:call-template>
</div>
```

Figure 5 : construction de l'indicateur "Evolution des capture déclarées par année et par nationalité de l'espèce (ici, le poulpe en Mauritanie).

La figure 6 présente une partie de celles-ci : version de l'indicateur, auteur, dernière date de mise à jour, complétude des données utilisées (définie sur une échelle de 1 à 5) et leur source (ici source 1).

```
<version><idversion>1</idversion></version>
<auteur>Ely</auteur>
<date_maj>2013-12-13</ date_maj>
<commentaires>Mise à jour à partir de données sources</commentaires>
<completude><ind_compl>5</ind_compl></ completude>
<source><idsource>1</ idsource ></ source>
```

Figure 6 : métadonnées associées à l'indicateur "Evolution des capture déclarées par année et par nationalité de l'espèce (ici, le poulpe en Mauritanie)

Les paramètres de l'indicateur diffèrent d'un indicateur à l'autre selon son type. Pour un indicateur graphique par exemple, (figure 7), on indique son type (idgraphe : histogramme, camembert, etc.), ses gabarits, ses marges.

```
<parametres>
<idgraph>6</idgraph>
<hauteur>300</hauteur>
<largeur>700</largeur>
<margebas>100</margebas>
<titrecourbe>Production annuelles</titrecourbe>
</parametres>
```

Figure 7 : définition des paramètres de l'indicateur

Les données sont définies par la requête qui sert à leur extraction et la connexion nécessaire (ici à la source si_mrt) définie dans un fichier de configuration (figure 8).

```
<dataxml>
<connection>si_mrt</connection>
<requete> select * from production </requete>
</dataxml>
```

Figure 8 : définition de l'extraction de données

4.4.2 Accès aux métadonnées des indicateurs d'une fiche thématique

L'accès à ces métadonnées est assuré à travers des icônes et les liens créés à cet effet et placés sous chaque indicateur (figure 9).

Pour une fiche donnée, à chaque indicateur, l'utilisateur peut accéder à la description complète des métadonnées de celui-ci, aux données qui ont servi à sa construction, à la description des sources initiales, etc.

4.4.3 Interaction - Collaboration.

Les expertises des professionnels peuvent venir expliciter et discuter les informations et synthèses produites au sein de l'atlas. Cet aspect n'a pas été traité dans cet article, mais il est bien pris en compte.

Au sein de chaque fiche, il est associé à chaque indicateur un fil de discussion (figure 10) permettant à chaque participant d'apporter des annotations (application Disqus [8]). Ces fils de discussion sont archivés et interviennent dans la phase de validation des indicateurs, avant ouverture au grand public.

[8] http://www.disqus.com/

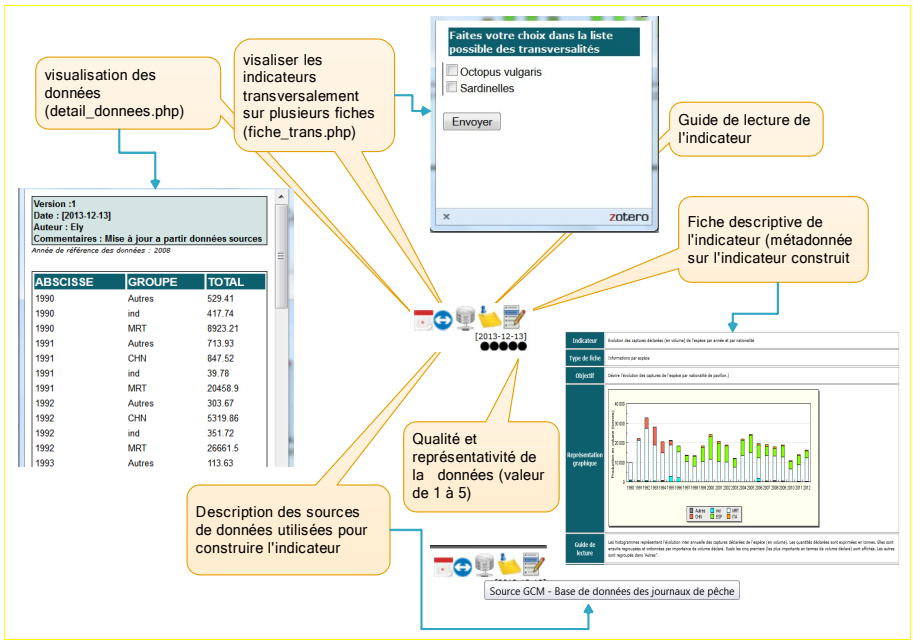


Figure 9 : accès aux métadonnées des indicateurs créés au sein de la fiche

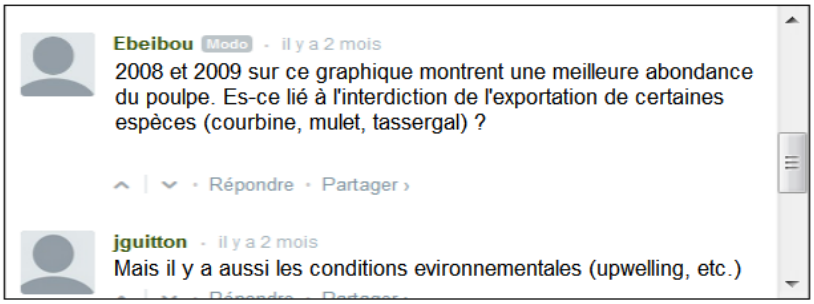


Figure 10 : exemple de fil de discussion (ici sur le poulpe mauritanien)

4.4.4 Les Web services d'accès aux données traitées

Certaines catégories d'utilisateurs voudront disposer des données brutes ou traitées, mais en dehors du contexte des fiches thématiques, pour pouvoir les injecter dans d'autres applications clientes et produire des sorties personnalisées répondant à des besoins spécifiques de recherche. Ces utilisateurs sont souvent des scientifiques ayant des besoins spécifiques et familiarisés avec des outils particuliers. Les applications clientes qu'ils voudront utiliser vont varier selon la discipline et le

profil de uns et des autres : SIG (Arcgis, Mapinfo, Grass, QGIS, gvSIG, ...), ou logiciels statistiques (Excel, R, …), etc.

A cet effet, notre système utilise des techniques simples (NuSOAP, WSDL et PHP) pour implémenter des services web d'accès aux données sous plusieurs formats (textes, XML, HML etc.). Pour l'accès aux couches géographiques, on envisage d'implémenter un serveur WMS autour de l'architecture Mapserver.

4.5 Travaux liés

Plusieurs systèmes ont été développés pour intégrer les sources d'information disparates et hétérogènes, dans cette zone du monde. Ils ont abordé le problème de deux manières différentes. La première consistait à intégrer physiquement les sources de données originelles. Cette méthode n'a pas fonctionné pour les raisons détaillées dans la section 2. La deuxième met en œuvre l'approche d'entreposage (SIAP[9], ISTAM[10], MDST[11]). Notre méthode est inspirée de cette approche.

5. Conclusion et perspectives

Nous avons dans cet article montré la conception et la première mise en œuvre d'un système d'information innovant, simulant la vision intégrée au service d'une thématique globale liée à l'approche écosystémique des pêches, conçue à partir de différents systèmes d'information locaux.

L'outil proposé permet la mise en relation à la fois des divers systèmes d'information et des experts autour d'un produit commun partagé (et non monopolisé par un seul acteur).

Dans la première phase, le prototype d'atlas en ligne propose deux produits distincts : i) un outil de gestion de connaissance et de synthèse des informations sous la forme d'un site web permettant l'accès dynamique aux fiches thématiques par soit des gestionnaires, soit du grand public ; ii) un accès normalisé à des indicateurs (texte, numérique, formes) et aux données source afférentes au travers de services web. Cet outil n'est accessible qu'à des utilisateurs référencés (par un module d'administration non détaillé au niveau de cet article).

Les perspectives consistent à poursuivre la conception des autres applications clientes : forum et module d'analyse spatiale. De plus au-delà des interactions physiques (permettant de franchir l'obstacle de l'interopérabilité syntaxique), nous mettrons l'accent sur la collaboration entre acteurs et l'interopérabilité sémantique (usage de vocabulaires contrôlés).

L'aspect collaboratif portera dans un premier temps principalement sur *le partage des données*, la *définition des indicateurs* qui devra être réalisée en

[9] Système d'Information et d'Analyse des Pêches
[10] Improve Scientific and Technical Advices for Management
[11] Model And Data Sharing Tool

concertation entre les différents acteurs et sur *l'expertise*, apportée à l'aide des commentaires qui seront associés à chaque indicateurs et qui viendront contextualiser les fiches de présentation des indicateurs.

Bibliographie

Batita, Wided, Stéphane Roche, Yvan Bédard, et Claude Caron. 2012. « WikiSIG and collaborative GeoDesign. Towards a theoretical framework ». *Revue internationale de géomatique* 22 (2): 255-285. doi:10.3166/rig.22.255-285.

Ciobanu, D., S., Roche, T., Badard et C., Caron, « Du wiki au wikiSIG », *Geomatica*,vol. 61, n°4, 2007, p. 455-469.

Cowen, D. J., Shirley, W.L., and Jensen, J., "Collaborative GIS: A video-conferencing GIS for Decision Makers", *Proceedings of the International Conference on Geographic Information*, Lisbon, 1998. 8p.

Giacomo Rambaldi, Mike McCall, Daniel Weiner, Peter Mbile and Peter Kyem (2004). *Participatory GIS (PGIS),* http://www.iapad.org/participatory_gis.htm.

Laurini, Robert. 1998. « Groupware for urban planning: an introduction ». *Computers, Environment and Urban Systems* 22 (4): 317-333. doi:10.1016/S0198-9715(98)00029-5.

Laurini, R., S. Servigne, et T. Tanzi. 2001. « A primer ON TeleGeoProcessing and TeleGeoMonitoring ». *Computers, Environment and Urban Systems* 25 (3): 249−265. doi:10.1016/S0198-9715(00)00024-7.

Laurini, Robert. 1998. « Groupware for urban planning: an introduction ». *Computers, Environment and Urban Systems* 22 (4): 317−333. doi:10.1016/S0198-9715(98)00029-5.

Rinner, Claus. 2001. « Argumentation maps: GIS-based discussion support for on-line planning ». *Environment and Planning B: Planning and Design* 28 (6): 847−63. doi:10.1068/b2748t.

Rundensteiner, Elke A., Andreas Koeller, et Xin Zhang. 2000. « Maintaining Data Warehouses over Changing Information Sources ». *Commun. ACM* 43 (6): 57−62. doi:10.1145/336460.336475.

Sieber, Renee. 2006. « Public Participation Geographic Information Systems: A Literature Review and Framework ». *Annals of the Association of American Geographers* 96 (3): 491−507. doi:10.1111/j.1467-8306.2006.00702.x.

Wiederhold, G. 1992. « Mediators in the architecture of future information systems ». *Computer* 25 (3): 38−49. doi:10.1109/2.121508.

Session 5

Ingénierie des documents et des connaissances

Une représentation graphique des schémas XML pour l'enseignement

Emmanuel Desmontils

LINA - Université de Nantes,
2 rue de la Houssinière, BP92208,
44322 Nantes Cedex 03

emmanuel.desmontils@univ-nantes.fr

RÉSUMÉ. *XML est un (méta-)langage actuellement très utilisé. Dans le cadre des formations en informatique, il est indispensable d'initier les étudiants à ce langage et, surtout, à tout son éco-système. Nous avons mis au point un modèle permettant d'accompagner l'enseignement de XML. Il propose de représenter un schéma XML sous la forme d'un graphe mettant en valeur les caractéristiques structurelles des documents valides. Nous présentons dans cet article les différents éléments graphiques et les améliorations qu'il apporte à la modélisation de données en XML.*

ABSTRACT. *Currently, XML is a widely used language. In the context of computer science teaching, it is necessary to introduce students to this language and, especially, at its eco-system. We have developed a model to support the teaching of XML. We propose to represent an XML schema as a graph highlighting the structural characteristics of the valid documents. We present in this paper visual elements and the improvements it brings to data modeling in XML.*

MOTS-CLÉS : *XML, Représentation graphique, Schéma, DTD, XSD, Relax NG, Modèle hiérarchique.*

KEYWORDS: *XML, Schema, DTD, XSD, Relax NG, Graph, Hierarchical model.*

1. Motivation et objectifs

De nos jours, XML (Bray *et al.*, 1997)[1] prend une place importante dans les systèmes informatiques. Ce (méta-)langage est utilisé par exemple aussi bien pour l'échange de données entre Web services, pour le paramétrage d'applications ou pour mémoriser de façon pérenne des informations (par exemple à travers des bases de données XML (Bourret, 1999 ; Gardarin, 2002)).

La complexité des documents XML est extrêmement variable. Même si beaucoup de structures (appelées schémas) sont simples, il est important de bien maîtriser la modélisation de tels documents. Il est souvent utile de s'appuyer sur des méthodologies de conception connues comme UML ou Merise (Carlson, 2001 ; Routledge *et al.*, 2002 ; Gardarin, 2002 ; Desmontils, 2005 ; Lonjon et Thomasson, 2006). Cependant, ces méthodes ne sont pas vraiment satisfaisantes pour les données hiérarchiques.

De plus, pour exploiter ou produire des documents XML, un développeur doit être capable de bien appréhender les schémas. Cela lui permet de tirer profit au mieux de la structure hiérarchique à travers les API dédiées ou les langages adaptés. Nous avons constaté par ailleurs que les utilisateurs de XML n'exploitent pas toujours bien cette structure hiérarchique. Il est donc important d'introduire, dans la formation des développeurs, un enseignement sur XML et son éco-système. Cette formation doit comprendre en particulier :

– les différents langages de schéma (DTD, XSD, Relax NG[2]),

– les principales API de programmation (SAX, DOM[3]),

– les bases de données XML (eXist (Meier, 2003), etc.) et les langages de requête (XPath, XQuery, etc.[4]),

– les langages de transformation (XSLT[5]).

Durant les nombreuses années de formation à XML, nous avons constaté que, pour tous ces outils, le choix d'une représentation graphique permet de mieux appréhender la structure du document à exploiter ou à produire. Nous avons donc recherché une représentation pour mettre en évidence les caractéristiques structurelles du document à valider, en particulier la structure hiérarchique. N'étant pas satisfaits des outils classiques, nous nous sommes inspirés des outils graphiques utilisés pour représenter les modèles relationnels pour proposer notre propre modèle.

Afin d'illustrer notre modélisation, nous avons repris, parmi les nombreux exercices utilisant le modèle, un sujet donné aux étudiants du Master MIAGE de Nantes en octobre 2013. Il concerne la modélisation d'un service (simplifié) de films à la demande auquel est adossé un réseau social spécialisé. Le texte décrivant le contexte, le schéma associé ainsi que le graphe complet qui peut être produit à partir de ce schéma

1. Extensible Markup Language (XML) 1.0 : `http://www.w3.org/XML/`
2. XSD : `http://www.w3.org/XML/Schema` ; Relax NG : `http://relaxng.org/`
3. SAX : `http://www.saxproject.org/` ; DOM : `http://www.w3.org/DOM/`
4. `http://www.w3.org/XML/Query/`
5. `http://www.w3.org/Style/XSL/`

sont donnés en section 9[6]. Cet exemple et les illustrations qui suivent utilisent le langage de schéma originel pour XML : DTD. Cependant, l'utilisation de XSD ou de Relax NG ne feraient pas apparaître de différence notable, car nous attachons le plus d'importance à l'aspect structurel des schémas plutôt qu'au typage des informations.

Après avoir présenté les modèles existants (section 2), nous présenterons notre modélisation graphique pour les éléments (section 3), les attributs (section 4) et la structuration (section 5). Après une discussion sur ce modèle (section 6), nous conclurons par le ressenti des étudiants et quelques perspectives.

2. Modèles existants

Les outils de manipulation de schémas dédiés à XML proposent couramment des représentations graphiques pour les schémas XSD ou Relax NG. La figure 1 propose un extrait de représentation graphique typique de schéma XSD[7]. La représentation graphique de Relax NG (quand elle existe) est quasiment identique. Pour les DTD, il n'existe pas de représentation graphique dédiée. Ces représentations ont en commun une structuration sous forme de forêt d'arbres (horizontaux), avec la possibilité de déployer ou non certaines branches. Chaque arbre représente un concept du schéma. Un concept utilisé plusieurs fois apparaitra dans plusieurs branches. Ceci introduit une redondance visuelle importante pouvant amener à des schémas lourds à explorer ("ami" ou "mot-clé" dans notre exemple). Certains symboles sont utilisés sans nécessité, comme le symbole de séquence pour un seul élément (comme "prémium-standard" avec "liste-amis"). On y trouve aussi des symboles peu visibles (le '@' pour les attributs par exemple). La structure en graphe n'est pas clairement visible, le ou les éléments potentiellement racines ne sont pas identifiés et les liens de composition des éléments ne sont pas toujours très lisibles. Relativement faciles à implémenter, ils ne sont pas vraiment utilisables en dessin sur papier.

Les "Feature Models" (Kang *et al.*, 1990) proposent une modélisation hiérarchique spécifique qui a la propriété d'être formalisée (Schobbens *et al.*, 2006). Les symboles sont peu nombreux (4 pour le modèle de base), mais pas vraiment explicites, en particulier pour les cardinalités multiples qui peuvent être explicitées, mais seulement à l'aide de texte (ce qui n'est pas satisfaisant (Moody, 2009)).

Du point de vue général, il existe de nombreux outils graphiques de représentation des modèles conceptuels de données comme le paradigme Entité-Association-Propriété pour la méthode Merise (Tardieu *et al.*, 1983 ; Quang *et al.*, 1991), les diagrammes de classes pour le formalisme UML (Booch *et al.*, 1998), etc. Ces modèles permettent de modéliser n'importe quelle structure de données, mais, de ce fait, ne

6. Les figures de ce rapport ont été conçues à l'aide du logiciel de dessin vectoriel OmniGraffle `http://www.omnigroup.com/omnigraffle`.
7. Oxygen `http://www.oxygenxml.com/xml_editor/xml_schema_editor.html` est présenté ici. Des outils similaires sont présentées sur `http://en.wikipedia.org/wiki/XML_Schema_Editor`.

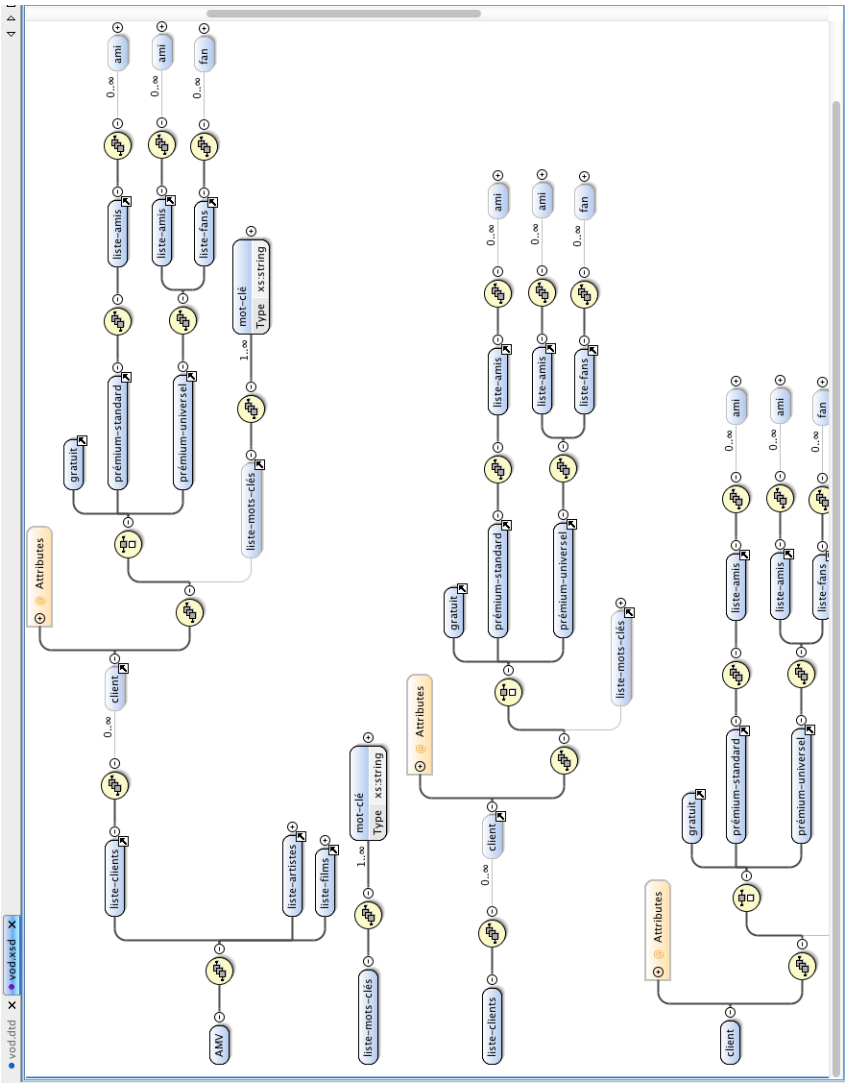


Figure 1. *Visualisation XSD avec Oxygen XML Editor 15.1*

sont pas nécessairement adaptés pour une compréhension des modèles hiérarchiques et ne sont pas faciles à appréhender graphiquement (Moody, 2009).

Parmi ces modèles standard, nous nous sommes intéressés aux "Crow's Foot Diagrams" (CFD) de (Everest, 1976). Ils sont utilisés depuis longtemps en ingénierie de l'information (Martin et Finkelstein, 1988) pour la représentation des tables et de leurs liens dans le modèle Entité-Relation. La forme graphique des cardinalités multiples (1..n ou 0..n) donne son nom à ce type de diagramme. Les CFD sont reconnus

pour leur expressivité visuelle (Moody, 2009) et sont assez familiers, car fréquemment utilisés dans les entreprises. La figure 2 présente un exemple de représentation d'un modèle relationnel avec la notation CFD. Ces pattes mettent en évidence de manière visuelle qu'un étudiant (resp. une matière) sera lié(e) à plusieurs notes.

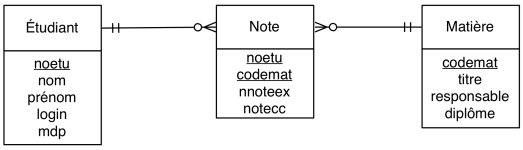


Figure 2. *Exemple de CFD pour le modèle relationnel*

Notre objectif est donc de proposer une représentation graphique pour l'initiation à XML, facile à mémoriser (en limitant le nombre de symboles), facile à comprendre (avec des notations intuitives, comme les CFD) et pouvant éventuellement être utilisée en travaux dirigés (avec papier et crayon) par des novices en informatique. Nous proposons donc un modèle indépendant du schéma XML (DTD, XSD, Relax NG, etc.) utilisant des formes identifiant clairement les concepts manipulés (utilisant des variables visuelles bien distinctes (Bertin, 1983 ; Moody, 2009)), en particulier en adoptant un codage redondant, au sens de (Moody, 2009), sur la forme et la couleur. Ceci permet ainsi d'avoir une représentation visuelle donnant une intuition de la structure hiérarchique des documents XML valides. Ces préoccupations se retrouvent dans (Bihanic *et al.*, 2013 ; Le Pallec et Dupuy-Chessa, 2013) de manière plus générale pour la modélisation du SI dans le contexte de l'ingénierie des modèles. Dans les sections 3, 4 et 5, nous allons détailler les différents composants du modèle.

3. Modélisation des éléments

La notion d'élément XML est centrale, car elle amène le vocabulaire du dialecte et sa grammaire. Dans cette première section, nous nous intéresserons uniquement à des éléments simples. Les éléments forment les sommets du graphe. Les liens entre les éléments correspondent aux contenus des éléments décrits par le schéma. Ces liens seront décrits en section 5.

Un élément est défini en DTD par < !ELEMENT nom-élément contenu>[8] où "contenu" est une expression s'apparentant aux expressions régulières et qui décrit la structure du contenu de l'élément. Dans notre modèle, un élément est représenté par un rectangle vert contenant le nom de l'élément. La figure 3a représente un élément vide décrit par < !ELEMENT like EMPTY>. Certains éléments n'ont pas de contenu structuré : leur contenu est le plus souvent textuel. Dans notre modèle, les textes seront représentés par un rectangle blanc. Éventuellement, ce rectangle contiendra un terme décrivant

8. Ici, comme dans la suite de ce document, les extraits de schéma en DTD seront présentés avec la police Courrier.

le type de texte contenu. La figure 3b représente un élément "like" avec un contenu textuel. Cet élément est décrit en DTD par < !ELEMENT like (#PCDATA)>.

(a) Élément vide (b) Élément à contenu textuel

Figure 3. *Élément*

Les éléments ne sont pas les seules sources d'information en XML : il y a aussi les attributs. Nous allons maintenant nous y intéresser.

4. Modélisation des attributs

XML autorise le positionnement d'attributs associés aux éléments. La liste des attributs est représentée par un rectangle arrondi jaune. Les attributs ont un type (CDATA, ID, etc.) et un comportement (#REQUIRED, #IMPLIED, etc.). La table 1 présente les cas d'attributs les plus fréquemment rencontrés et leur forme dans notre modèle.

	Modèle	**Traduction en DTD**
1	nom-cl	nom-cl CDATA #REQUIRED
2	%date-modif	date-modif CDATA #IMPLIED
3	pseudo	pseudo ID #REQUIRED
4	#client	client IDREF #REQUIRED
5	#(clients)	clients IDREFS #REQUIRED
6	{stars}	stars (0|1|2|3|4|5) #REQUIRED
7	stars/'0'	stars CDATA '0'

Tableau 1. *Formes d'attributs*

Notons que, dans le cas d'une liste de valeurs possibles (ligne 6), notre modèle est incomplet. Ce n'est pas très grave au regard des objectifs pédagogiques de notre modélisation. Cependant, pour être plus complet, il est possible d'écrire "stars∈{0,1,2,3,4,5}".

La figure 4a représente par exemple un élément vide décrit en DTD par < !ELEMENT client EMPTY>. Il possède trois attributs décrits par :

– <ATTLIST client pseudo ID #REQUIRED>,

– <ATTLIST client nom-cl CDATA #REQUIRED>,

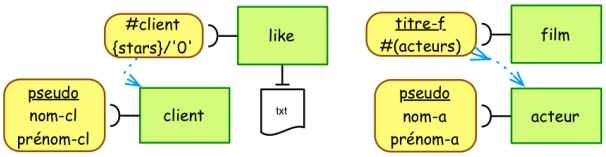
– <ATTLIST client prénom-cl CDATA #REQUIRED>.

Les cas de la table 1 peuvent être combinés. Par exemple, la figure 4b propose un attribut "{stars}/'0'" qui est un attribut pris dans une liste de valeurs (par exemple "0|1|2|3|4|5") et avec comme valeur par défaut '0'.

(a) avec un identifiant "pseudo" (b) avec une référence "client"

Figure 4. *Attributs et identifiants*

Afin de préciser le rôle des attributs de type IDREF(S), il est possible d'ajouter une flèche en pointillé allant de cet attribut vers l'identifiant qu'il est supposé référencer. Les schémas XML ne prévoient pas de préciser ce lien, mais il facilite l'exploitation des documents DTD ou XSD (mise au point des API, utilisation des langages de recherche d'information, etc.). La figure 5a montre le lien entre un attribut IDREF et l'attribut ID correspondant. Ici, l'attribut "client" de l'élément "like" doit contenir le "pseudo" d'un client. La figure 5b représente le lien entre un attribut IDREFS (ici "acteurs") et l'attribut ID qui correspond ("pseudo").



(a) ID ⟵ IDREF (b) ID ⟵ IDREFS

Figure 5. *Lien entre ID et IDREF(S)*

Ces liens ne font pas à proprement parler partie du graphe. Ils sont plus des commentaires, des appuis, pour les utilisateurs du graphe. Maintenant que les noeuds du graphe ont été modélisés, nous allons nous intéresser à la modélisation des arcs.

5. Modélisation des contenus complexes

La structure de graphe apparaît lorsqu'un élément en contient d'autres. Nous allons présenter les différents cas standard et élémentaires que nous pouvons rencontrer.

Tout d'abord, nous allons représenter les opérateurs d'itération "*", "+" et " ?". Pour cela, nous nous sommes inspirés des CFD dont l'aspect graphique est plus intuitif que les symboles. La forme de patte sous-entend bien la présence de l'élément en plusieurs exemplaires. Nous exploitons ici (et dans les autres représentations de cette section) les propriétés "physiques" des CFD, en particulier leur perception visuelle (Moody, 2009). La figure 6 présente la représentation utilisée pour chacun des opérateurs. Ces opérateurs permettent de mettre en place les arcs élémentaires entre les différents sommets du graphe.

Il reste deux opérateurs à introduire : la composition d'éléments en séquence et l'alternative. Chacun des éléments qui composent la séquence ou l'alternative font

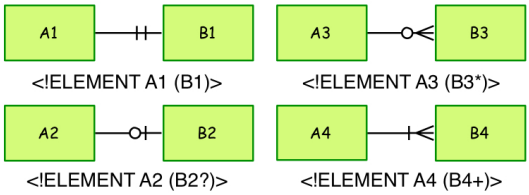
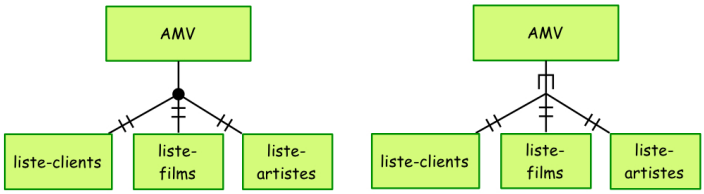


Figure 6. *Itération d'éléments*

l'objet d'un des opérateurs d'itération, d'une autre séquence, d'une autre alternative ou d'un sous-groupe.

La séquence permet de décrire un contenu comme une suite ordonnée d'éléments. Elle est représentée dans notre modèle par un point. L'ordre des nœuds est l'ordre dans le parcours trigonométrique autour de ce point en partant de la gauche du modèle. La figure 7a illustre une séquence et se traduit par :
< !ELEMENT AMV (liste-clients, liste-films, liste-artistes)>.



(a) séquence (b) alternative

Figure 7. *Composition d'éléments*

L'alternative permet de donner le choix entre plusieurs éléments. Elle est représentée dans notre modèle par une fourche. La figure 7b illustre une alternative et se traduit par < !ELEMENT AMV (liste-client | liste-films | liste-artistes)>.

Certaines descriptions de contenu sont construites en utilisant des sous-groupes. Par exemple, supposons qu'un artiste puisse avoir été acteur dans certains films, metteur en scène dans d'autres, voire compositeur de bande originale. Alors, nous pourrions proposer la définition suivante : < !ELEMENT artiste (joue | réalise | compose)+>. La partie (joue | réalise | compose) dans l'exemple ci-dessus est un sous-groupe sur lequel est appliqué l'opérateur d'itération "+". Les trois éléments sont alors répétés dans un ordre quelconque (alternative vue précédemment). Dans notre modèle, un sous-groupe est mis en évidence par un pentagone beige à bords oranges. La figure 8 illustre l'exemple ci-dessus. Cette notion de sous-groupe est "récursive" : un sous-groupe peut lui-même contenir des sous-groupes. Dans notre modèle, il y aura alors une imbrication de zones.

Pour terminer, il est parfois utile d'importer le schéma d'un dialecte spécifique ou d'un langage standard. Dans ce cas, il n'est pas toujours utile de le détailler (impor-

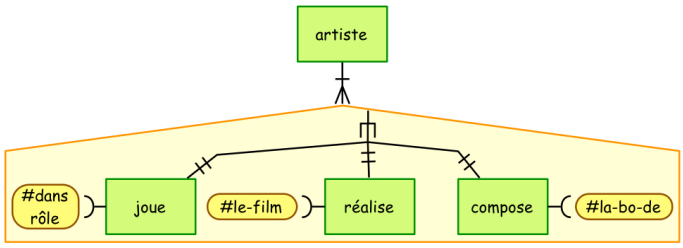
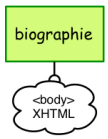


Figure 8. *Sous-groupe d'éléments*

tance secondaire ou, au contraire, parfaitement maîtrisé). Dans ce contexte, il n'est pas nécessaire d'expliciter son graphe. Aussi, nous avons introduit dans notre modèle un symbole représentant un sous-graphe qui n'est pas détaillé. Pour cela, nous utilisons le symbole du nuage. La figure 9a représente la biographie d'un artiste en XHTML[9]. Ce langage, bien connu, n'a pas besoin d'être représenté pour être manipulé. Il suffit alors de donner à "biographie" le même contenu que la balise "<body>" en XHTML (contenu qui est décrit par l'entité paramètre "%body ;").



(a) contenu identique à celui de <body> en XHTML

(b) contenu spécifique, mais non-détaillé pour le moment

Figure 9. *Contenu non-détaillé d'un élément*

Cette simplification peut aussi permettre de ne présenter qu'un graphe partiel. La partie mise en ellipse peut être considérée comme sous-entendue. Le graphe principal est alors considéré comme un graphe hiérarchique, la partie sous-entendue peut faire l'objet d'un graphe ultérieur. La figure 9b représente le contenu de l'élément "liste-films" comme une partie du graphe non développée. Cela permet de réduire la complexité visuelle du graphe et d'introduire la modularisation (Moody, 2009).

L'organisation des éléments dans la page est importante (variables plantaires de (Bertin, 1983)). La disposition des éléments doit, quand c'est possible, rappeler la structure arborescente des documents valides. L'axe vertical permet de représenter la filiation alors que l'axe horizontal représente la fratrie. L'élément supposé racine se trouve en haut du document.

9. La DTD de XHTML est décrite à l'URL : `http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd`.

6. Discussion

Ce modèle a été mis au point à des fins pédagogiques. Il n'est donc pas adapté à des schémas très complexes que l'on peut trouver dans certains systèmes d'information. En effet, la topographie du graphe devient difficile à aborder sur une seule page A4 avec un grand nombre d'éléments ou lorsque le nombre des imbrications de groupes dépasse deux ou trois niveaux. La complexité du diagramme devient un frein à sa compréhension. Notre exemple complet (section 9) illustre bien cette difficulté. En effet, il ne comporte qu'une vingtaine d'éléments et semble, visuellement, déjà bien complexe. Par contre, il reste utilisable en conditions opérationnelles pour de petits schémas, souvent présents dans des applications Web à base de services par exemple.

Nous pouvons noter aussi que le graphe seul reste incomplet pour une vue exhaustive des éléments et des attributs. La description du contenu des attributs et des éléments reste très basique. Le modèle ne couvre même pas l'ensemble des types en DTD : les listes ne sont pas détaillées, les types NMTOKEN et NMTOKENS ne sont pas identifiés, tout comme le type NOTATION. Les entités générales ne sont pas non plus explicitées. Par contre, les entités paramètres le sont de manière indirecte (par leurs effets). Dans notre exemple (section 9), le mécanisme d'inclusion de la DTD de XHTML s'effectue par ce principe.

Malgré tout, ces schémas restent un appui de taille pour aborder la construction de chemins de localisation en XPath, pour comprendre le mécanisme de parcours par l'API SAX, les optimisations de parcours avec DOM ou l'application des règles en XSLT. En XPath, par exemple, il est plus facile de construire le chemin pour atteindre une information en se référençant à notre modèle. En particulier, les liens entre les attributs de type IDREF ou IDREFS (une originalité de notre modèle) et leur correspondant de type ID permettent de mieux comprendre les fonctions XPath "id()" (en suivant la flèche) et "idref()" (en remontant la flèche).

De plus, l'utilisation des symboles tirés des CFD permet de donner une idée intuitive de la structure de l'arbre XML résultant. Pour renforcer l'intuition, il est conseillé de travailler aussi sur la topographie du graphe. En effet, dans la plupart des cas, le graphe est déjà un arbre, voir un treillis (comme dans notre exemple en 9), et il est utile de présenter le graphe comme tel. Notons que le cas d'un treillis est intéressant, car il permet de visualiser simplement les éléments pouvant se trouver dans différents contextes (cas de "liste-mots-clés" et "liste-amis" dans notre exemple) en limitant la redondance d'objets que l'on constate dans le modèle classique.

Notre modélisation en graphe peut s'inscrire dans le processus de modélisation de données ou de connaissances sous forme hiérarchique, comme l'illustre la figure 10.

En effet, un processus de modélisation standard commence par l'analyse du problème. Cette analyse amène la construction d'un modèle conceptuel des données, en Merise ou en UML par exemple, puis à un modèle physique, dans notre contexte en XML (Carlson, 2001 ; Routledge *et al.*, 2002 ; Gardarin, 2002 ; Desmontils, 2005 ; Lonjon et Thomasson, 2006) (étapes 1, 2 et 4 dans la figure 10). Notre modèle peut

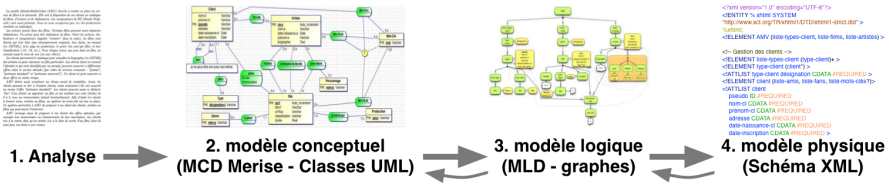


Figure 10. *Processus de modélisation avec notre modèle*

s'insérer entre le modèle conceptuel et le schéma (étape 3 dans la figure 10). En effet, il peut être considéré comme le modèle logique des données, car il va mettre en valeur les propriétés du modèle physique XML (structure hiérarchique et contrôle syntaxique) et réaliser certains types de contraintes énoncées dans le modèle conceptuel. Ainsi, comme dans notre exemple, il est possible de profiter de listes de valeurs courtes (les types d'abonnement par exemple) pour les transformer en contraintes structurelles. De même, les contraintes du modèle conceptuel peuvent parfois être transformées en contraintes syntaxiques dans le modèle logique (Desmontils, 2005). Dans notre exemple, la liste d'amis n'est possible que dans les abonnements "prémium".

Dans le cadre pédagogique uniquement, le processus de modélisation est alors pris à rebours en partant du schéma, objet de l'étude, pour remonter vers la modélisation en graphe pour mettre en évidence la structure.

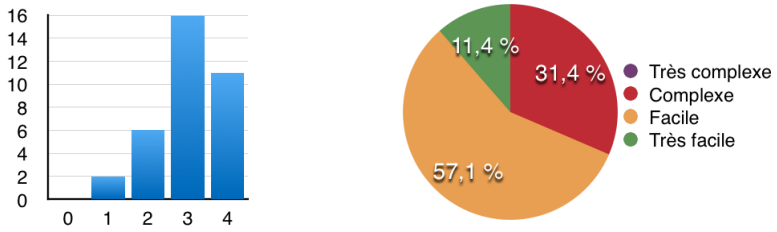
7. Conclusion et perspectives

Le modèle graphique de schémas XML que nous présentons ici est donc un modèle orienté vers l'aspect structurel du document XML et moins vers l'aspect type de données. C'est un modèle intéressant pour l'enseignement, surtout si le schéma n'est pas trop complexe. Il permet une bonne maîtrise du schéma, surtout pour la découverte de XPath, XSLT et des API de programmation (SAX et DOM). Il peut aussi être utilisé lors du processus de construction du schéma XML en jouant le rôle de modèle logique des données (entre le modèle conceptuel en Merise ou UML et le modèle physique qu'est XML).

Le modèle est opérationnel dans le cadre des enseignements de l'éco-système XML en Master MIAGE à l'université de Nantes et en Master CCI depuis quelques années. Cette année, nous avons demandé aux étudiants ayant suivi notre enseignement sur XML (deux modules respectivement au premier et au second semestre) d'évaluer l'apport de ce type de graphe (figure 11)[10]. Notons que ces étudiants n'ont pas eu de présentation détaillée du modèle et que les graphes proposés n'avaient pas de légende. Les concepts ont été présentés au fur et à mesure. Pour les différents exemples et exer-

10. Le détail des réponses à ce sondage peut être consulté à l'URL suivante :
`http://www.desmontils.net/Documents/G4LX/G4LX-2014-03-28.xlsx`

cices donnés en CM, TD, TP et contrôles continus, nous donnions systématiquement le graphe, le schéma (souvent la DTD) et un exemple de document XML valide.



Utilité du modèle (0 = "N'apporte rien" ; 4 = "Très utile") Maitrise du modèle

Figure 11. *Sondage auprès des étudiants de MIAGE et de CCI de Nantes*

Sur les 52 étudiants, 35 ont répondu (soit plus de 67%). Nous leur avons demandé l'utilité de ce modèle (figure 11a). Ils devaient mettre une note entre 0 (n'apporte rien) et 4 (très utile). 77% ont trouvé le modèle utile (notes 3 et 4) et ils ont donné une note moyenne de 3,02. Nous leur avons aussi demandé la complexité d'apprentissage de ces graphes (figure 11b). Pour 68,6% des étudiants, ils ont été faciles à appréhender. Certains ont trouvé ce type de graphe complexe, en partie à cause du manque d'explications initiales (indiqué dans les commentaires). Les objectifs pédagogiques sont donc en grande partie atteints. Plusieurs pistes d'améliorations sont encore à explorer.

Tout d'abord, nous l'avons déjà évoqué, notre modèle graphique ne permet pas de représenter tous les concepts présents dans un schéma, en particulier pour XSD. Il serait intéressant de mieux préciser graphiquement les caractéristiques (types) des attributs et des contenus textuels sans pour autant augmenter de manière exagérée la complexité visuelle. En particulier, il manque un modèle graphique pour représenter les types complexes et les opérations d'extension et de restriction. Il ne faut cependant pas oublier que l'objectif principal n'est pas de proposer un modèle graphique exhaustif, mais de s'attacher aux propriétés structurelles du schéma à des fins pédagogiques.

Ensuite, à plusieurs reprises, nous avons évoqué la difficulté à présenter un modèle graphique complet et exploitable sans dégrader exagérément la complexité visuelle. Il sera donc intéressant de proposer un cadre général d'analyse permettant de comparer les modèles et de mesurer leurs limites (comme le nombre d'éléments au delà duquel la complexité visuelle sera trop dégradée) en s'appuyant sur les travaux de (Moody, 2009 ; Le Pallec et Dupuy-Chessa, 2013). Ceci permettra d'apporter une aide pour la modularisation des représentations graphiques de schémas XML, mais aussi d'évaluer les schémas eux-mêmes, par exemple en proposant une estimation de l'intérêt pédagogique d'un exercice au regard de la complexité graphique.

Enfin, nous avons montré que notre modèle graphique peut s'insérer dans un processus global de modélisation. Il convient maintenant de mettre en place des processus (semi-)automatiques permettant de passer du modèle physique (schéma) à notre modèle, et réciproquement. De même, il faudra étudier des techniques (semi-)automatiques permettant de passer du modèle conceptuel (MCD Merise ou classes UML) à notre

modèle. Pour cela, nous pourrons évidemment exploiter les travaux effectués en ingénierie des modèles (IDM) (Sendall et Kozaczynski, 2003).

Remerciements

Nous tenons à remercier les étudiants de la MIAGE et du Master CCI de Nantes d'avoir servi de cobaye pour ce travail depuis plusieurs années. Nous voulons aussi remercier P. André, C. Attiogbé et A. Mostéfaoui pour leur soutien et à J.-M. Mottu pour ses relectures et ses conseils. Ce travail a été partiellement financé par le projet ONECAD (Fondation de France, Université de Nantes -Géolittomer UMR 6554 & LINA UMR C6241-).

8. Bibliographie

Bertin J., « Semiology of graphics : Diagrams, networks, maps (WJ Berg, Trans.) », *Madison, WI : The University of Wisconsin Press, Ltd*, , 1983.

Bihanic D., Chevalier M., Dupuy-Chessa S., Morineau T., Polacsek T., Le Pallec X., « Modélisation graphique des SI : Du traitement visuel de modèles complexes », *Actes de la Conférence Inforsid 2013*, mai 2013.

Booch G., Rumbaugh J., Jacobson I., « The Unified Modeling Language (UML) », *World Wide Web : http://www.rational.com/uml/ (UML Resource Center)*, vol. 94, 1998.

Bourret R., « XML and Databases », 1999.

Bray T., Paoli J., Sperberg-McQueen C. M., Maler E., Yergeau F., « Extensible Markup Language (XML) », *World Wide Web Journal*, vol. 2, n° 4, 1997, p. 27–66.

Carlson D. A., *Modeling XML applications with UML : practical e-business applications*, Addison-Wesley Reading, 2001.

Desmontils E., « Modélisation avec XML, Cours e-miage », 2005, in French, http://miage.univ-nantes.fr/miage/D2X1/chapitre_modelisationXML/chapitre.htm.

Everest G. C., « Basic data structure models explained with a common example », *Proc. Fifth Texas Conference on Computing Systems*, Austin, TX, october 1976, IEEE Computer Society publications office, p. 18–19.

Gardarin G., *XML des bases de données aux services Web*, Dunod, 2002, in French.

Kang K. C., Cohen S. G., Hess J. A., Novak W. E., Peterson A. S., « Feature-oriented domain analysis (FODA) feasibility study », rapport, 1990, DTIC Document.

Le Pallec X., Dupuy-Chessa S., « Support for quality metrics in metamodelling », *Proceedings of the Second Workshop on Graphical Modeling Language Development*, ACM, 2013, p. 23–31.

Lonjon A., Thomasson J.-J., *Modélisation XML*, Editions Eyrolles, 2006, in French.

Martin J., Finkelstein C., *Information Engineering*, Savant Institute, Jan 1988, http ://miha.ef.uni-lj.si/predmeti/informatika/Information_engineering.pdf.

Meier W., « eXist : An open source native XML database », *Web, Web-Services, and Database Systems*, p. 169–183, Springer, 2003, http ://exist-db.org.

Moody D. L., « The "physics" of notations : toward a scientific basis for constructing visual notations in software engineering », *Software Engineering, IEEE Transactions on*, vol. 35, n° 6, 2009, p. 756–779, IEEE.

Quang P. T., Chartier-Kastler C., Davison D., *MERISE in Practice*, Macmillan, 1991.

Routledge N., Bird L., Goodchild A., « UML and XML schema », *Australian Computer Science Communications*, vol. 24, n° 2, 2002, p. 157–166.

Schobbens P., Heymans P., Trigaux J.-C., « Feature diagrams : A survey and a formal semantics », *Requirements Engineering, 14th IEEE int. conf.*, IEEE, 2006, p. 139–148.

Sendall S., Kozaczynski W., « Model transformation : The heart and soul of model-driven software development », *IEEE software*, vol. 20, n° 5, 2003, p. 42–45.

Tardieu H., Rochfeld A., Colletti R., Lesourne J., *La méthode MERISE : principes et outils*, Editions d'organisation, 1983, in French.

9. Description de l'exemple

En octobre 2013, les étudiants de Master MIAGE 1ère année ont eu à modéliser en XML les données sur le projet suivant :

La société AtlanticMobileVideo (AMV) cherche à mettre en place un service de films à la demande. Elle met à disposition de ses clients un catalogue de films, d'acteurs et de réalisateurs. Les compositeurs de BO (Bande Originale) sont aussi présents. Nous ne nous occuperons pas, ici, des producteurs (sociétés ou individus). Les acteurs jouent dans des films. Certains films peuvent avoir plusieurs réalisateurs. Un acteur peut être réalisateur de films. Outre les acteurs, réalisateurs et compositeurs (appelés "artistes" dans la suite), les films sont décrits par leur titre (pas nécessairement original), leur durée, un résumé (en XHTML), le(s) pays de production, le genre (un seul par film) et leur classification (-12, -16, etc.). Pour chaque acteur qui joue dans un film, on connait aussi le nom de son (ou ses) rôle(s). Les clients parcourent le catalogue pour consulter la biographie (en XHTML) des artistes ou pour visionner un film particulier. Les clients (dont on connait l'identité et qui sont identifiés par un pseudo) peuvent souscrire à différentes offres selon le service attendu (par ordre de services croissant : "gratuit", "prémium standard" et "prémium universel"). Un client ne peut souscrire à deux offres en même temps. AMV désire aussi constituer un réseau social de cinéphiles. Aussi, les clients peuvent :

– se lier à d'autres clients avec l'offre "prémium standard" ou "prémium universel" ;

– se déclarer "fan" d'un artiste ou apprécier un film en lui mettant une note (étoile) de 0 à 5, avec un commentaire textuel éventuellement, seulement avec l'offre "prémium universel".

Afin d'aider les clients à trouver amis, artistes ou films, un système de mots-clés est mis en place. Ce système permettra à AMV de proposer à un client des clients, artistes ou films qui pourraient l'intéresser. AMV envisage aussi de proposer à ses clients des offres spéciales, par exemple leur anniversaire ou l'anniversaire de leur inscription. Les clients nés à la même date qu'un artiste (ou à la date de sortie d'un film) dont ils sont fans ont droit à une remise.

On peut obtenir alors le schéma (DTD) suivant :

```
<!ENTITY % xhtml SYSTEM
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd" > %xhtml;
<!ELEMENT AMV (liste-types-client,liste-films,liste-artistes)>
```

```
<!-- -- Gestion des clients -- -->
<!ELEMENT liste-types-client (client)*>
<!ELEMENT client ((gratuit|prémium-standard|prémium-universel),
                                liste-mots-clés?)>
  <!ATTLIST client pseudo ID #REQUIRED
    nom-cl CDATA #REQUIRED prénom-cl CDATA #REQUIRED
    adresse CDATA #REQUIRED
    date-naissance-cl CDATA #REQUIRED
    date-inscription CDATA #REQUIRED>
<!ELEMENT gratuit EMPTY >
<!ELEMENT prémium-standard (liste-amis)>
<!ELEMENT prémium-universel (liste-amis, liste-fans)>
<!ELEMENT liste-amis (ami*)>
<!ELEMENT ami EMPTY> <!ATTLIST ami avec IDREF #REQUIRED>
<!ELEMENT liste-fans (fan*)>
<!ELEMENT fan EMPTY> <!ATTLIST fan de IDREF #REQUIRED>

<!-- -- Gestion des films -- -->
<!ELEMENT liste-films (genre)+> <!ELEMENT genre (film+)>
  <!ATTLIST genre désignation CDATA #REQUIRED>
<!ELEMENT film (résumé,liste-likes,produit+,liste-mots-clés?)>
  <!ATTLIST film no-f ID #REQUIRED
    titre-f CDATA #REQUIRED classification (0|10|12|16|18) '0'
    date-sortie CDATA #REQUIRED durée CDATA #REQUIRED
    URL-flux CDATA #REQUIRED>
<!ELEMENT résumé (%block;)> <!--%block; contenu de <boby>-->
<!ELEMENT liste-likes (like*)> <!ELEMENT like (#PCDATA)>
  <!ATTLIST like client IDREF #REQUIRED
    date-like CDATA #REQUIRED stars (0|1|2|3|4|5) '0'>
<!ELEMENT produit EMPTY> <!ATTLIST produit pays CDATA #REQUIRED>

<!-- -- Gestion des artistes -- -->
<!ELEMENT liste-artistes (artiste)+>
<!ELEMENT artiste
   (biographie, liste-mots-clés?, (joue|réalise|compose)+)>
  <!ATTLIST artiste no-a ID #REQUIRED
    nom-a CDATA #REQUIRED prénom-a CDATA #IMPLIED
    nationalité CDATA #REQUIRED date-naissance-a CDATA #REQUIRED>
<!ELEMENT biographie (%block;)>
<!ELEMENT joue EMPTY>
  <!ATTLIST joue dans IDREF #REQUIRED rôle CDATA #REQUIRED>
<!ELEMENT réalise EMPTY> <!ATTLIST réalise le-film IDREF #REQUIRED>
<!ELEMENT compose EMPTY> <!ATTLIST compose la-bo-de IDREF #REQUIRED>

<!-- -- Eléments communs -- -->
<!ELEMENT liste-mots-clés (mot-clé+)> <!ELEMENT mot-clé (#PCDATA)>
```

La figure 12 présente un exemple de graphe associé à notre exemple.

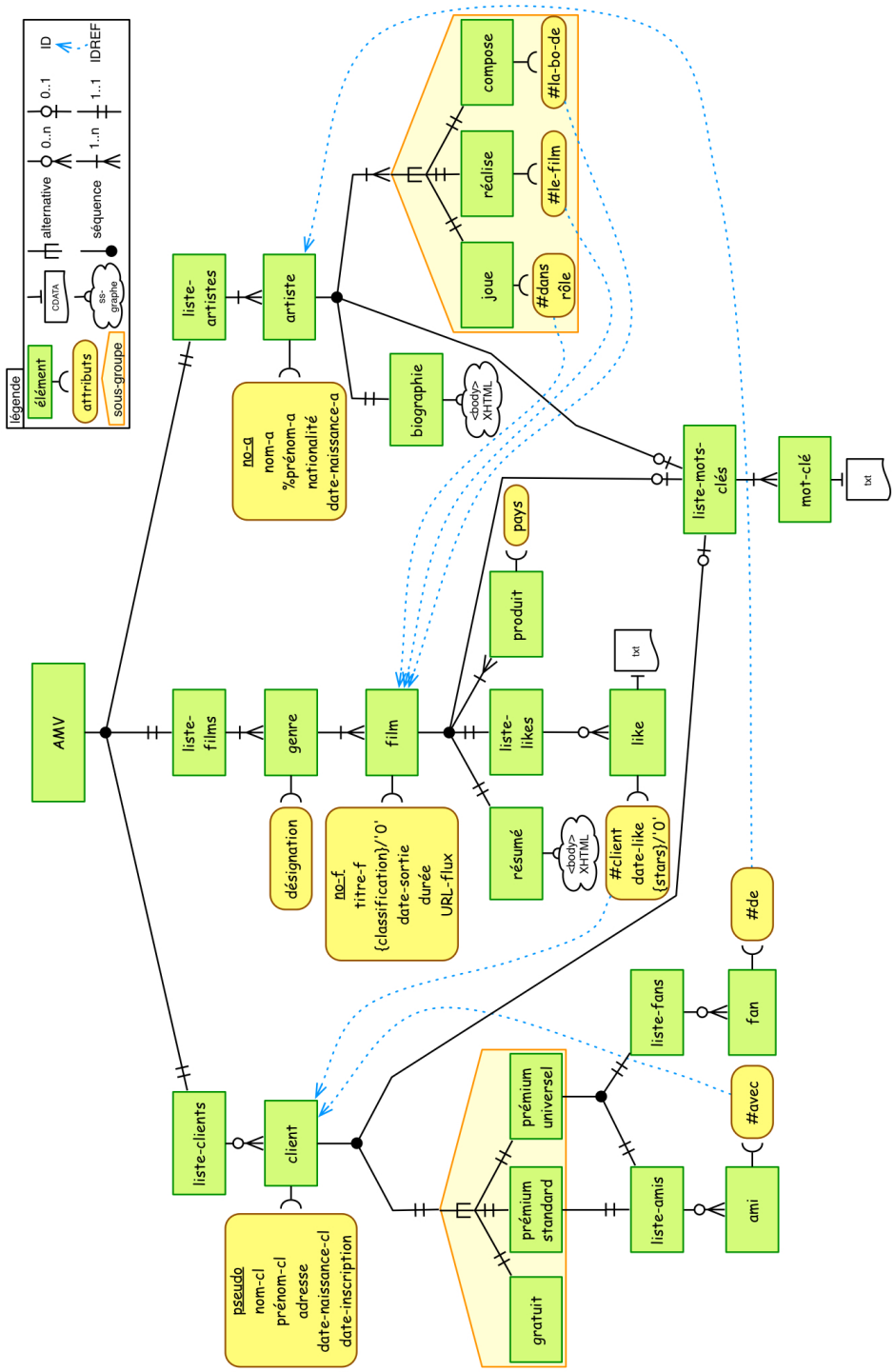


Figure 12. *AMV*

Prise en compte des préférences des utilisateurs pour l'estimation de la pertinence multidimensionnelle d'un document

Bilel Moulahi[*,**] — **Lynda Tamine**[*] — **Sadok Ben Yahia**[**,***]

[*] *Université Paul Sabatier, Institut de Recherche en Informatique de Toulouse, France*
bilel.moulahi@irit.fr, lynda.tamine@irit.fr

[**] *Université de Tunis El Manar, Faculté des Sciences de Tunis, LIPAH, 2092, Tunisie*
sadok.benyahia@fst.rnu.tn

[***] *Institut Mines-Télécom, Télécom SudParis, UMR CNRS Samovar, 91011 Evry Cedex, France*

RÉSUMÉ. *Dans ce papier, nous proposons une nouvelle approche d'agrégation personnalisée pour l'estimation de la pertinence multidimensionnelle. L'approche est basée sur un opérateur d'agrégation mathématique qui utilise une mesure floue permettant la quantification de l'importance estimée des critères pour chaque utilisateur ainsi que leur degré d'interactivité ou d'interdépendance. Nous évaluons l'opérateur d'agrégation proposé en utilisant la collection de test standard fournie avec par la tâche "Contextual Suggestion" de TREC* 2013*. Les résultats expérimentaux obtenus montrent l'impact de la personnalisation sur les performances de recherche.*

ABSTRACT. *In this paper, we propose a novel personalized aggregation approach to the multidimensional relevance aggregation. The approach is based on a mathematical aggregation operator relying on a fuzzy measure that allow to quantify the importance degree of each relevance dimension for every user as well as the interaction existing between the criteria. The evaluation of our approach is performed on the test collection of TREC* 2013 *Contextual Suggestion track. Experimental results show the impact of the personnalisation of search results.*

MOTS-CLÉS : *Personnalisation, préférences, pertinence, Choquet personnalisé, capacité.*

KEYWORDS: *Pesonalization, preferences, relevance, personalized Choquet, capacity.*

1. Introduction

De nombreux travaux en recherche d'information (RI) ont mis en exergue à la fois l'importance et la complexité du concept "*pertinence*" (Borlund, 2003, Saracevic, 2007, Taylor *et al.*, 2007). Son importance est liée au fait que la notion sous-jacente est le fondement des modèles d'ordonnancement de documents en réponse à une requête, qui est la finalité même d'un système de RI (Baeza-Yates *et al.*, 1999). Sa complexité est, quant à elle, subordonnée à deux propriétés. La première concerne la multiplicité de ses dimensions, vues comme des ensembles de critères, qui peuvent être de surcroît, interdépendantes ; même si de nombreux travaux du domaine se sont focalisés sur la dimension thématique seule, force est de constater que de nombreux autres travaux ont prouvé empiriquement l'impact conjoint de plusieurs dimensions sur l'estimation de la pertinence finale, comme la tâche et la situation de recherche (Borlund, 2003, Saracevic, 2007, Taylor *et al.*, 2007). Considérons à titre d'exemple, une tâche de recherche de *tweets* ; des analyses expérimentales ont montré que la pertinence d'un *tweet* en réponse à une requête, est impactée principalement par la conjonction de trois dimensions qui sont le sujet et la fraîcheur du *tweet* et l'autorité du *tweeterer* qui l'a émis (Nagmoti *et al.*, 2010). La seconde propriété concerne la subjectivité qui entoure ces dimensions ; en effet, la plupart d'entre elles ne sont pas basées sur des estimations objectives puisqu'elles sont fortement liées à la perception personnelle des utilisateurs impliqués dans la tâche de RI ; on cite à titre d'exemple les centres d'intérêt, l'expertise et les préférences des utilisateurs. La problématique scientifique est alors de définir des opérateurs capables d'agréger des scores de pertinence partiels (relatifs à chaque dimension) en tenant compte de leur interdépendance éventuelle. Cette problématique a été abordée dans diverses applications de RI comme la RI personnalisée (Sieg *et al.*, 2007, Tamine *et al.*, 2006, Daoud *et al.*, 2010), la RI mobile (Göker *et al.*, 2008), la RI sociale (Nagmoti *et al.*, 2010) et la RI géographique (Mata *et al.*, 2011). Cependant, ces travaux applicatifs ont généralement utilisé des opérateurs de calcul de moyenne pondérée ou de combinaison linéaire qui se basent sur l'hypothèse non réaliste d'additivité ou d'indépendance des dimensions. D'autres travaux fondamentaux récents, se sont intéressés en revanche à la définition d'opérateurs d'agrégation, indépendamment du cadre applicatif, qui permettent de traiter peu ou prou le biais de l'interactivité (da Costa Pereira *et al.*, 2012, Gerani *et al.*, 2012, Eickhoff *et al.*, 2013). Toutefois, ces opérateurs ne permettent pas de tenir compte de la propriété de subjectivité qui peut se décliner à travers les différences entre les utilisateurs quant à l'importance accordée à chaque dimension de pertinence. Notre contribution, présentée dans ce papier, répond à cet objectif. Plus précisément, nous proposons un opérateur flou d'agrégation basé sur l'intégrale de Choquet (Choquet, 1953, Grabisch, 1995), capable d'agréger des scores de pertinence personnalisés, puisque pondérés par l'importance orientée-utilisateur de chaque dimension.

La suite du papier est organisée comme suit : la section 2 présente un aperçu des travaux du domaine et situe notre contribution dans ce contexte. La section 3 détaille les principes de l'opérateur d'agrégation ainsi que l'algorithme d'apprentissage des mesures d'importance. La section 4 décrit le cadre expérimental puis les résultats

de l'application de l'opérateur proposé dans une tâche TREC dédiée à une tâche de RI personnalisée en l'occurrence "TREC[1] Contextual Suggestion" (Dean-Hall *et al.*, 2013).

1.1. *Synthèse des travaux*

Le concept de pertinence est incontestablement au centre d'une activité de recherche d'information comme en témoignent les nombreux travaux qui en ont fait l'objet d'étude (Saracevic, 1976, Borlund, 2003, Saracevic, 2007). L'un des résultats phares qui ressort de ces travaux est que la pertinence est estimée en globalité selon un ensemble de dimensions qui s'apparentent à des familles de critères ; parmi ces différentes dimensions, on cite les plus reconnues dont : la pertinence thématique (contenu et méta-contenu), la pertinence situationnelle (temps et géolocalisation) et la pertinence cognitive (expertise, centres d'intérêts). Un autre résultat important est l'interdépendance de ces dimensions pour inférer la pertinence globale d'un document (Nagmoti *et al.*, 2010, Saracevic, 2007). En clair, un utilisateur juge de la pertinence d'un document en tenant compte conjointement de l'ensemble des critères de pertinence ; à titre d'exemple, un document est d'autant plus pertinent du point de vue du contenu que l'expertise de l'utilisateur est en lien avec ce contenu. Historiquement, la dimension thématique est particulièrement considérée dans le domaine. La prise en compte de la propriété de multiplicité des dimensions de pertinence a particulièrement émergé dans des cadres applicatifs de la RI comme :

– la RI mobile (Göker *et al.*, 2008) : un document est d'autant plus pertinent pour une requête qu'il en est proche thématiquement et qu'il comporte des liens vers des lieux géographiquement proches de l'utilisateur qui est en situation de mobilité ;

– la RI sociale (Nagmoti *et al.*, 2010) : un document (ou ressource sociale) est d'autant plus pertinent pour une requête qu'il en est proche thématiquement, qu'il émane d'un acteur socialement important et qu'il est recommandé par un ami ;

– la RI personnalisée (Sieg *et al.*, 2007, Daoud *et al.*, 2007, Daoud *et al.*, 2010) : un document est d'autant plus pertinent pour une requête qu'il en est proche thématiquement et qu'il est en adéquation avec les centres d'intérêts de l'utilisateur ;

– RI géographique (Daoud *et al.*, 2013) : un document est d'autant plus pertinent pour une requête qu'il en est proche thématiquement et qu'il comporte des liens vers des lieux géographiquement proches des lieux cités dans la requête ;

La plupart de ces travaux exploitent des opérateurs classiques de produit, de moyenne pondérée et de combinaison linéaire. D'autres travaux (Palacio *et al.*, 2010) exploitent des opérateurs de combinaison inspirés de la fusion des données. Cependant ces opérateurs répondent à la problématique de l'agrégation en se basant sur l'hypothèse d'additivité ou d'indépendance des dimensions de pertinence. D'autres travaux récents ont particulièrement examiné le principe d'agrégation de dimensions interac-

1. `http://trec.nist.gov`

tives indépendamment du cadre applicatif (da Costa Pereira *et al.,* 2012, Gerani *et al.,* 2012, Eickhoff *et al.,* 2013). Celia et *al.* (da Costa Pereira *et al.,* 2012) ont proposé un opérateur d'agrégation multidimensionnelle mettant en jeu quatre (4) critères des pertinence : contenu, couverture, adéquation et fiabilité en définissant deux opérateurs d'agrégation prioritaire en l'occurrence, "*And*" et "*Scoring*". Ces opérateurs modélisent un ordre de priorité entre les critères de pertinence sur la base d'un mode de calcul de poids associés qui favorise la satisfaction du critère d'ordre supérieur ; les travaux présentés dans (Bouidghaghen *et al.,* 2011) ont montré l'efficacité de ces opérateurs dans un cadre de RI mobile. Gerani et *al.* (Gerani *et al.,* 2012) ont proposé un opérateur qui ne nécessite pas la satisfaction de la condition de comparabilité des scores partiels de pertinence. Ils utilisent à cet effet un algorithme de transformation de scores basé sur l'algorithme *Alternating Conditional Expectation* et le modèle *Box-Cox*. Plus récemment, Eickhoff et *al.* (Eickhoff *et al.,* 2013) ont proposé une approche statistique basé sur la méthode *Copulas* qui traite spécifiquement la complexité des dépendances des critères de pertinence. Eickhoff et *al.* ont montré que la méthode *Copulas* permet de modéliser des relations de dépendances complexes entre les différentes dimensions de pertinence. Leur approche a été évaluée dans trois tâche de RI à savoir, la recherche d'opinions dans les blogs, la recherche personnalisée dans les folksnonomies et la recherche web adaptées aux enfants.

1.2. *Aperçu de la contribution et positionnement*

La cadre général de nos travaux concerne l'agrégation de dimensions de pertinence, qu'elles soient interdépendantes ou indépendantes, dans une tâche de RI. Plus spécifiquement, nous présentons une approche d'agrégation personnalisée des scores de pertinence basée sur l'usage d'une mesure floue, appelée capacité, sous-jacente à l'opérateur de Choquet (Choquet, 1953). Cette mesure est à la base de la quantification de l'importance estimée de chaque dimension pour chaque utilisateur ainsi que leur degré d'interactivité ou d'interdépendance ; elle est estimée selon un algorithme d'apprentissage, qui infère les mesures optimales en utilisant une vérité de terrain évaluable à l'aide de la métrique précision de la recherche. Nous évaluons l'opérateur d'agrégation proposé en utilisant la collection de test standard TREC Contextual Suggestion (Dean-Hall *et al.,* 2013) et montrons l'impact de la prise en compte des dépendances entre critères ainsi que leur personnalisation sur les performances de recherche. Comparativement aux travaux antérieurs proches (da Costa Pereira *et al.,* 2012, Gerani *et al.,* 2012, Eickhoff *et al.,* 2013) ainsi qu'à notre précédente contribution (Moulahi *et al.,* 2013), le travail présenté dans ce papier s'en distingue selon les principaux points clés suivants :

1) une agrégation pondérée par les préférences des utilisateurs quant à chacune des dimensions agrégées, contrairement aux travaux de l'état de l'art présentés dans (da Costa Pereira *et al.,* 2012, Gerani *et al.,* 2012, Eickhoff *et al.,* 2013) ainsi que dans notre précédente contribution (Moulahi *et al.,* 2013) ; ces travaux proposent de déployer des opérateurs produisant des scores de pertinence dépendant seulement des

dimensions de pertinence agrégées, indépendamment des utilisateurs ;

2) un nouvel algorithme d'apprentissages des mesures d'importance des critères, comparativement à l'algorithme présenté dans (Moulahi *et al.*, 2013) ;

3) une nouvelle évaluation expérimentale tant dans l'objectif que dans la méthodologie, qui montre, comparativement à celle menée dans (Moulahi *et al.*, 2013), à la fois l'intérêt de l'agrégation et de la personnalisation des préférences des utilisateurs sur les performances de recherche.

2. Agrégation personnalisée de la pertinence multidimensionnelle

2.1. *Formalisation de l'opérateur d'agrégation*

Nous introduisons le problème d'agrégation de pertinence multidimensionnelle comme étant un problème de prise de décision multicritères où les critères considérés sont les dimensions de pertinence. En effet, le défi majeur dans le problème d'agrégation est : (1) l'estimation de l'importance des critères : identifier les critères devant avoir un poids d'importance plus élevé que d'autres ; (2) l'agrégation : combiner efficacement les critères de pertinence en tenant compte des dépendances pouvant exister entre eux.
Soient $\mathcal{D} = \{d_1, d_2, \ldots, d_M\}$ un ensemble de documents, $\mathcal{C} = \{c_1, c_2, \ldots, c_N\}$ l'ensemble des critères de pertinence et q une requête donnée. La tâche de combinaison des critères notée $RSV^u_{c_i}(q,d_j)$, d'un document $d_j \in \mathcal{D}$, obtenu suivant chaque critère de pertinence $c_i \in \mathcal{C}$, est appelé *agrégation*. La fonction $\mathcal{F}$ qui calcule le score de pertinence personalisé du document d_j en réponse à la requête q, d'un utilisateur u, a la forme suivante :

$$\mathcal{F} : \begin{cases} \mathbb{R}^N \longrightarrow \mathbb{R} \\ (RSV^u_{c_1}(q,d_j) \times \ldots \times RSV^u_{c_N}(q,d_j)) \longrightarrow \mathcal{F}(RSV^u_{c_1}(q,d_j), \ldots, RSV^u_{c_N}(q,d_j)) \end{cases}$$

Où $RSV^u_{c_i}(q,d_j)$ est le score de pertinence de d_j suivant le critère c_i, étant donné l'utilisateur u.

Dans ce qui suit, nous allons nous baser sur l'intégrale de Choquet comme un opérateur d'agrégation de pertinence multidimensionnelle. Cette fonction mathématique est construite à l'aide d'une mesure floue (ou *capacité*) μ, définie comme suit.

Définition 1 *Soit $I_\mathcal{C}$ l'ensemble de tous les sous ensembles de critère de $\mathcal{C}$. Une mesure floue est une fonction monotone normalisée μ de $I_\mathcal{C}$ à $[0 \ldots 1]$ tels que :*
$\forall I_{C_1}, I_{C_2} \in I_\mathcal{C}$, *si* $(I_{C_1} \subseteq I_{C_2})$ *alors* $\mu(I_{C_1}) \leq \mu(I_{C_2})$, *avec* $\mu(I_\varnothing) = 0$ *et* $\mu(I_\mathcal{C}) = 1$.

Pour simplifier la notation, $\mu(I_{C_i})$ sera dénotée par μ_{C_i}. La valeur de μ_{C_1} peut être interprétée par le degré d'importance de l'interaction entre les critères inclus dans le sous ensemble C_1. La fonction d'agrégation de pertinence personnalisée basée sur l'intégrale de Choquet est définie comme suit :

Définition 2 *$RSV^u_{\mathcal{C}}(q, d_j)$ est le score de pertinence personnalisé de d_j pour l'utilisateur u suivant l'ensemble des critères de pertinence $\mathcal{C} = \{c_1, c_2, \ldots, c_N\}$ défini comme :* $RSV^u_{\mathcal{C}}(q, d_j) = Ch_\mu(RSV^u_{c_1}(q, d_j), \ldots, RSV^u_{c_N}(q, d_j))$
$= \sum_{i=1}^{N} \mu^u_{\{c_i,\ldots,c_N\}}.(rsv^u_{(i)j} - rsv^u_{(i-1)j})$

Où Ch_μ la fonction d'agrégation de Choquet, $rsv^u_{(i)j}$ est le $i^{\text{ème}}$ élément de la permutation $RSV(q, d_j)$ sur le critère c_i, tel que ($0 \leq rsv^u_{(1)j} \leq ... \leq rsv^u_{(N)j}$), $\mu^u_{\{c_i,\ldots,c_N\}}$ est le degré d'importance de l'ensemble des critères $\{c_i, ..., c_N\}$ pour l'utilisateur u.
De cette manière, nous sommes capables d'ajuster les paramètres du modèle d'ordonnancement automatiquement pour chaque utilisateur, rendant ainsi les résultats dépendants de ses préférences sur les critères considérés. Notons que si μ est une mesure additive, l'intégrale de Choquet correspond à la moyenne pondérée. Sinon, elle demande moins de 2^N mesures de capacité dans le cas où la mesure est est $k-$additive, *i.e.,* $\mu_A = 0$ pour tous les sous ensembles de critères $A \subseteq \mathcal{C}$ avec $|A| > k$. D'un point de vue théorique, l'intégrale de Choquet dispose d'un nombre de propriétés qui semblent être pertinentes pour un domaine tel que la RI ; étant donné qu'elle est construite à partir du concept de mesure floue, elle permet la modélisation des relations d'interaction flexibles en considérant des relations de dépendance complexes entre les critères (Grabisch *et al.,* 2000).
Pour faciliter la tâche d'interprétation du modèle résultat de l'intégrale de Choquet, nous allons exploiter deux paramètres appelés, "indice d'importance" et "indice d'interaction" (Grabisch *et al.,* 2000). L'indice d'importance, appelé également indice de Shapley, permet d'estimer la contribution moyenne qu'un critère (c_i) apporte à toutes les autres combinaisons de critères possibles. L'indice d'interaction permet de donner des informations sur le phénomène d'interaction pouvant exister entre un ensemble de critères. Pour des détails sur le calcul de ces deux indices, le lecteur peut se référer au papier original (Grabisch *et al.,* 2000).

2.2. *Apprentissage des préférences des utilisateurs*

L'objectif de la phase d'apprentissage est d'optimiser les mesures floues selon une mesure objective de RI (e.g. $P@X$) en identifiant les valeurs de capacité permettant de personnaliser les résultats de recherche d'un utilisateur en particulier, tout en considérant ses préférences individuelles sur les critères de pertinence.

Nous proposons dans ce qui suit un algorithme générique permettant d'apprendre ces capacités indépendamment du nombre de critères de pertinence, et de la tâche de RI considérée. Étant donné un utilisateur, les données d'apprentissage pour identifier les mesures floues de l'intégrale de Choquet comprennent un ensemble de requêtes d'apprentissage, et pour chaque requête, un ensemble ordonné de documents représentés par des vecteurs contenant des scores partiels selon chaque critère ; chaque

Notation	Description
Q^u_{app}	L'ensembe des requêtes utilisées pour apprendre les valeurs de capacités de l'utilisateur u
N	Nombre de critères de pertinence
$\mathcal{D}$	La collection de documents
K	Nombre de documents utilisés pour l'apprentisage pour chaque requête
$\gamma^{i,r}$	Liste ordonnée de documents en réponse à la requête q_r suivant la combinaison de capacité $\mu^{(i)}$. Soit $P@X(\gamma^{r,i})$ la $P@X$ de $\gamma^{r,i}$ et $AVP@X(\gamma^i)$ soit sa moyenne de $P@X$ sur toutes les requêtes $\in Q_{app}$ suivant $\mu^{(i)}$
I_{Cr}	Tous les sous ensembles de critères possibles de Cr
$\mathcal{S}_\mu$	Ensemble de combinaisons de capacité expérimentées. Chaque combinaison $\mu^{(i)} \in \mathcal{S}_\mu$ contient les valeurs de capacités de tous les ensembles et sous ensemble de critères

Tableau 1 – Synthèse des notations utilisées avec l'algorithme 1.

document est annoté avec une étiquette (*e.g.,* pertinent ou non pertinent). La méthodologie adoptée est détaillée dans l'algorithme 1. Le tableau 1 décrit les notations utilisées dans cet algorithme. Ce dernier comprend deux étapes principales :

– *Initialisation des valeurs initiales des combinaisons de capacités.* Une combinaison de capacités $\mu^{(.)}$ désigne l'ensemble des valeurs de capacités associées à chaque critère et à chaque sous-ensemble de critères. Par exemple, dans le cas de trois critères de pertinence, une combinaison de capacités comprend ($\{\mu_{c_1}; \mu_{c_2}; \mu_{c_3}; \mu_{c_1,c_2}; \mu_{c_1,c_3}; \mu_{c_2,c_3}\}$). Afin de paramétrer ces valeurs, nous utilisons une mesure de RI telle que la $P@X$ sur les requêtes d'apprentissage Q^u_{app}. Le paramétrage est concevable étant donné que le nombre de critères de pertinence est généralement petit (Saracevic, 2007). Cependant, lorsque le nombre de critères est supérieur ou égale à 4, nous pouvons éviter la complexité du paramétrage en se basant sur la famille des capacités 2-additive (Grabisch *et al.,* 2000) nécessitant moins de coefficients à définir.

– *Optimisation des valeurs de capacités.* En partant d'une combinaison de capacités $\mu^{(*)}$ obtenue dans l'étape précédente, on extrait les K premiers documents retournés en réponse à chaque requête $q \in Q^u_{app}$. Les scores de ces documents (D^u_{learn}) sont interpolés pour placer les documents non pertinents à la fin de l'ordonnancement. Après avoir obtenu les scores de pertinence globaux désirés $RSV^{int}_{\mathcal{C}}(q, d_j)$ pour chaque document $d_j \in D^u_{app}$, et étant donné que nous disposons des étiquettes $RSV^u_{c_i}(q, d_j)$, nous procédons à l'application de la méthode des moindres carrées pour l'identification des valeurs de capacités des critères et des sous-ensembles de critères considérés.

Algorithm 1 Apprentissage des mesures floues

Entrées: Q^u_{learn}, N, K.
Sortie: Combinaison de capacité optimale $\mu^{(**)}$.

Étape 1 : Initialisation des valeurs de capacités

$m \leftarrow (1-N) \times N$;

1. **Pour** $i = 1$ à m {*Identification des combinaisons de capacités*} **Faire**
2. $\mu^{(i)} = (\bigcup_{j:1..N} \{\mu_{c_j}\}) \bigcup (\bigcup_{Cr \in \mathcal{C}, |Cr|>1} \{\mu_{I_{Cr}}\})$; $\mu_{I_{Cr}} = \sum_{c_i \in Cr, |c_i=1|} \mu_{c_i}$
3. **Fin Pour**
4. **Si** $N \geq 4$ {*Supposer la* 2*-additivité*} **Alors**
5. **Pour** chaque $I_{Cr} \in \mu^{(i)}$ tel que $|Cr| > 2$ **Faire**
6. $\mu_{I_{Cr}} = 0$
7. **Fin Pour**
8. **Fin Si**
9. $\mathcal{S}_\mu = \bigcup_{i:1..m} \{\mu^{(i)}\}$
10. **Pour** chaque $\mu^{(i)} \in \mathcal{S}_\mu$ {*paramétrage des capacités*} **Faire**
11. Calculer $AVP@X(\gamma^i)$
12. **Fin Pour**
13. $Cmax = \underset{1...|\mathcal{S}_\mu|}{\text{Argmax}} (AVP@X(\gamma^i))$; $\mu^{(*)} = \mu^{(cmax)}$

Étape 2 : Optimiser les valeurs de capacités

14. $D^u_{app} = \emptyset$
15. **Pour** $r = 1$ à $|Q^u_{app}|$ {*Interpoler les scores globaux*} **Faire**
16. $D^u_{app} = D^u_{app} \cup \gamma^{*,r}$
17. **Pour** $j = 1$ à K **Faire**
18. $RSV^{int}_{\mathcal{C}}(q_r, d_j) = \underset{1...d'_j \in \gamma^{*,r}, d'_j \succ_{\mathcal{C}} d_j}{\text{Max}} (RSV^u_{\mathcal{C}}(q_r, d'_j))$; $\gamma^{*,r} = \gamma^{*,r} \setminus \{d_j\}$
19. **Fin Pour**
20. **Fin Pour**

{*Optimisation basée sur la méthode des moindres carrées*}

21. **Répéter**

$$\mathcal{F}_{LS}(\mu) = \sum_{d_j \in D^u_{learn}} [Ch_\mu(RSV^u_{c_1}(d_j), \ldots, RSV^u_{c_N}(d_j)) - RSV^{int}_{\mathcal{C}}(d_j)]^2$$

22. **Jusqu'à** convergence
23. **Retourner** le résultat $\mu^{(**)}$

3. Cadre expérimental

Notre évaluation expérimentale est basée sur la collection de test standard fournie par la tâche "Contextual Suggestion" de TREC[2] 2013 (Dean-Hall *et al.*, 2013). Cette tâche a pour objectif d'évaluer les techniques de recherche répondant à des besoins en information, qui sont fortement tributaires du contexte et des centres d'intérêts des uti-

2. Text REtrieval Conference (http ://trec.nist.gov/)

lisateurs. Étant donné un utilisateur, cette tâche a pour objectif de chercher les places d'attractions pouvant l'intéresser suivant deux critères de pertinence dépendants : (1) les centres d'intérêt de l'utilisateur, *i.e.*, ses préférences personnelles sur un historique de recherche de places ; (2) sa localisation géographique.

3.1. *Données expérimentales*

La collection de test présente les caractéristiques suivantes :

– **Utilisateurs :** le nombre total d'utilisateurs est égal à 635. Chaque utilisateur est représenté par un profil reflétant ses préférences sur des lieux d'une liste de 50 exemples de suggestions. Un exemple de suggestion est un lieu d'attraction qui est susceptible d'intéresser l'utilisateur. Chaque exemple est représenté par le titre du lieu, une brève description et une URL du site web correspondant. Les préférences des utilisateurs sont données sur une échelle de 5 points et sont attribuées aux descriptions et aux URLs des exemples de suggestions. Les préférences positives (*resp.*, négatives) sont celles ayant un degré de pertinence égal à 3 ou à 4 (*resp.*, 0 ou 1) selon la description du site et la correspondance par rapport à l'URL.

– **Contextes et requêtes :** le nombre de contextes fournis est égal à 50 ; chaque contexte correspond à une position géographique dans une ville donnée. La position géographique est décrite par une longitude et une latitude. Étant donnée une paire d'utilisateurs et un contexte représentant la requête, l'objectif principal de la tâche est de fournir une liste de 50 suggestions triée par ordre de pertinence selon les critères centres d'intérêt de l'utilisateur et géolocalisation.

– **Collection de documents :** pour chercher des suggestions de lieux à partir du web, nous avons exploité l'API Google Place[3]. Comme pour la plupart des groupes participant à la tâche "Contextual Suggestion" (Dean-Hall *et al.*, 2013), nous commençons par interroger l'API Google Place avec les requêtes appropriées en se basant sur la localisation géographique des lieux. Étant donné que l'API Google Place renvoie jusqu'à 60 suggestions par requête, nous avons effectué une nouvelle recherche avec des paramètres différents tels que les types de lieux qui sont pertinents par rapport à la tâche (*e.g.*, restaurant, pizzeria, musée, etc.). Nous avons collecté, en moyenne, environ 157 suggestions par requête et 3925 suggestions au total. Pour obtenir les scores des documents collectés selon le critère de géolocalisation, nous avons calculé la distance entre les lieux collectés et le contexte. Les scores des documents selon le critère centres d'intérêts est calculé en se basant sur le cosinus de similarité entre la description des suggestions et le profil de l'utilisateur. Les profils des utilisateurs sont représentés par des vecteurs de termes construits à partir de leurs préférences personnelles sur les exemples de suggestions. La description des lieux est construite à partir des "snippets" des résultats renvoyés par le moteur de recherche Google[4] lorsque l'URL du lieu est soumise sous forme d'une requête.

3. `https://developers.google.com/places`
4. `https://www.google.com`

– **Jugements de pertinence :** les jugements de pertinence de cette tâche sont effectués par les utilisateurs et mandatés par TREC à la fois (Dean-Hall *et al.*, 2013). Chaque utilisateur représenté par un profil, juge les lieux qui lui sont suggérés de la même façon que les exemples de suggestions. Ainsi, l'utilisateur affecte un jugement de $0-4$ à chaque titre/description et à chaque URL, tandis que les assesseurs de TREC jugent les suggestions uniquement en termes de correspondance au critère géolocalisation avec une évaluation de (2, 1 et 0). Une suggestion est considérée comme pertinente si elle a un degré de pertinence égal à 3 ou 4 selon le critère centre d'intérêts (profil) et une évaluation égale à 1 ou 2 selon le critère géolocalisation. Dans ce qui suit, ces jugements de pertinence constituent notre réalité de terrain utilisée pour l'apprentissage et le test.

3.2. *Protocole d'évaluation*

Nous avons adopté une méthodologie entièrement automatisée basée sur une validation croisée afin d'identifier les valeurs de capacité des utilisateurs et tester les performances du modèle d'agrégation. À cette fin, nous avons procédé à une partition aléatoire de l'ensemble des 50 contextes en deux ensembles de même taille, noté Q^u_{app} et Q^u_{test} utilisés respectivement pour l'apprentissage et le test. En outre, pour éviter le problème de surapprentissage, l'ensemble des contextes est divisé aléatoirement dans un second tour en deux ensembles différents d'apprentissage et de test.

L'objectif principal de la phase d'apprentissage est d'apprendre les capacités ($\mu^u_{\{centre\_interet\}}$, $\mu^u_{\{localisation\}}$) qui correspondent à l'importance des critères de pertinence. Nous commençons d'abord par une mesure floue initiale donnant le même poids d'importance pour les deux critères de pertinence. Ensuite, nous calculons la mesure de précision $P@5$ de tous les contextes de l'ensemble d'apprentissage Q^u_{app}. En utilisant la vérité de terrain fournie avec la tâche "Contextual Suggestion" de TREC 2013, et en se basant sur l'algorithme 1, nous identifions pour chaque utilisateur ses préférences personnelles sur les deux critères : centres d'intérêts et localisation géographique. Enfin, pour tester l'efficacité de notre approche, nous nous sommes appuyés sur l'ensemble de contextes restants Q^u_{test} et nous avons utilisé la mesure officielle de la tâche $P@5$ pour le calcul de performances. Cette mesure de précision est équivalente à la proportion des suggestions de lieux pertinents retournés parmi les 5 premiers.

4. Résultats expérimentaux

4.1. *Analyse de l'importance des critères de pertinence*

Notre premier objectif consiste à analyser les valeurs de capacité issues de l'algorithme 1, représentant le degré d'importance des critères de pertinence pour les utilisateurs ($\mu^u_{\{centre\_interet\}}$, $\mu^u_{\{geolocalisation\}}$). A cet effet, nous commençons par analyser l'importance intrinsèque de chaque critère indépendamment des autres cri-

tères. La figure 1 montre la variation des valeurs de capacité pour chaque utilisateur selon les deux critères de pertinence sur l'ensemble Q^u_{app} d'apprentissage. L'axe des abscisses représente l'ensemble des utilisateurs (35-669) et l'axe des ordonnées représente les valeurs de capacité correspondantes selon les critères centres d'intérêt (Ci) et géolocalisation (Geo).

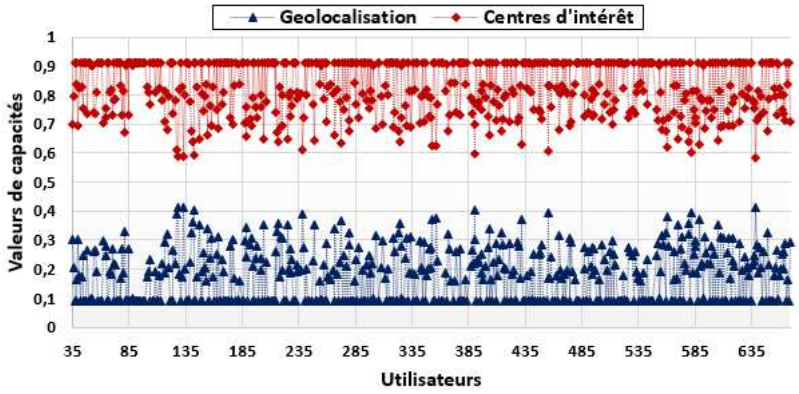


Figure 1 – Valeurs de capacités des utilisateurs de la tâche "Contextual Suggestion" de TREC 2013 suivant les deux critères de pertinence centres d'intérêt et géolocalisation.

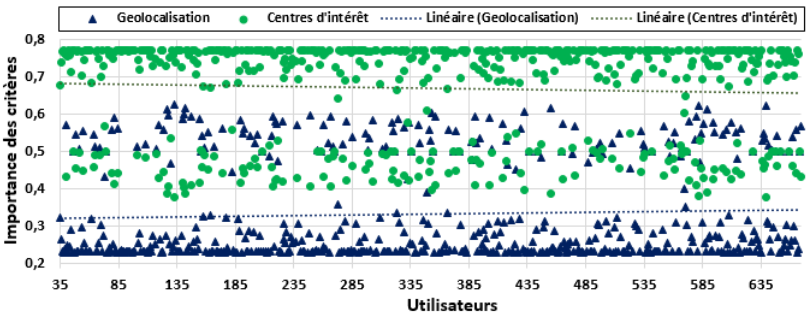


Figure 2 – Importance des critères centres d'intérêt (Ci) et géolocalisation pour les utilisateurs de la tâche "Contextual Suggestion" de TREC 2013.

En se référant à la figure 1, nous constatons que le critère Ci se voit accorder une capacité plus importante que le critère Geo. Par exemple, l'utilisateur 285 a une valeur de capacité de l'ordre de $0,23$ pour le premier critère alors qu'il a une mesure de l'ordre de $0,76$ pour le critère Geo. Ceci est prévisible étant donné que les utilisateurs de cette tâche s'intéressent généralement aux lieux qui correspondent à leurs préférences personnelles, même si elles ne sont pas géographiquement pertinentes. Cependant, la figure 1 montre que la distribution des valeurs de capacité est loin d'être la même pour tous les utilisateurs et met en exergue des valeurs qui vont de $0,09$ à $0,414$ pour le critère Geo et d'autres qui vont de $0,585$ à $0,909$ pour le critère Ci.

Pour mieux comprendre ce constat, nous traçons sur la figure 2, les valeurs des indices d'importance reflétant, pour chaque utilisateur, le degré de préférence globale selon les deux critères de pertinence Ci et Geo. A la différence de la figure 1, la figure 2 met en évidence l'importance moyenne de chaque critère de pertinence quand il est associé à l'autre critère. On peut observer sur la figure 2 que les préférences des utilisateurs sur les deux critères sont totalement différentes. Le lissage des valeurs d'importance obtenues selon ces critères donne deux courbes linéaires avec des valeurs tout à fait constantes et différentes, corroborant ainsi les résultats obtenus sur la figure 1. Le critère "centre d'intérêt" est encore pondéré par une importance relativement élevée pour la plupart des utilisateurs. Néanmoins, on peut également remarquer au milieu de la figure (valeurs comprises entre $0,4$ et $0,7$) que certains utilisateurs ont une préférence élevée sur le critère géolocalisation et inversement.

Dans une seconde étape, nous analysons à travers la figure 3, la dépendance entre les critères pour chaque utilisateur par le biais de l'indice d'interaction (Grabisch, 1995). Plus les valeurs de cet indice sont proches de 1 (*resp.*, -1) plus les deux critères sont dépendants et l'interaction est positive (*resp.*, négative). Si la valeur de l'indice d'interaction est égale 0, les deux critères sont considérés comme indépendants et par conséquent, il n'existe aucune interaction entre ces derniers. On peut constater que les valeurs obtenues sur tous les utilisateurs sont toutes positives et varient entre $0,28$ et $0,99$. La valeur moyenne est de l'ordre de $0,56$ ce qui implique une interaction positive entre les deux critères de pertinence considérés lorsqu'ils sont combinés ensemble.

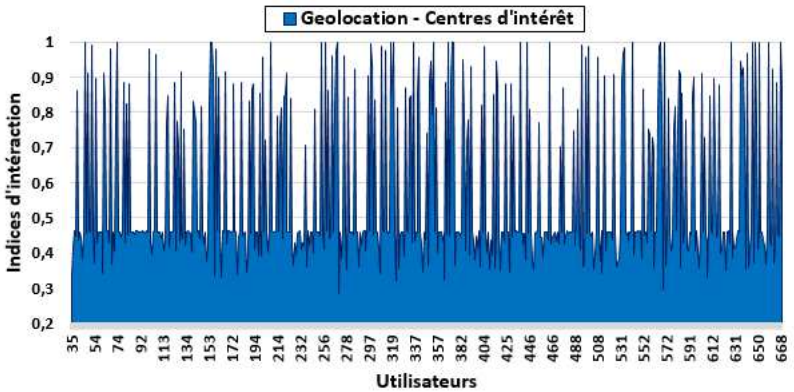


Figure 3 – Indices d'intéraction entre les critères de pertinence centres d'intérêt et géolocalisation pour chaque utilisateur.

4.2. *Analyse des performances de recherche*

Notre second objectif est d'évaluer les performances de notre approche en termes : (i) d'agrégation de pertinence multidimensionnelle ; et (ii) de personnalisation des

préférences des utilisateurs sur les critères de pertinence. Pour ce faire, nous comparons les résultats obtenus sur l'ensemble de contextes de test Q^u_{test} aux méthodes d'agrégation de référence (*baseline)* : la moyenne arithmétique pondérée (MAP) largement utilisée dans la plupart des approches impliquant la combinaison des scores de pertinence et les deux opérateurs d'agrégation prioritaires SCORING et AND, précédemment utilisés pour l'agrégation de pertinence dans un cadre de RI personnalisée. Il convient de préciser que nous avons effectué une série d'expérimentations avec une validation croisée pour identifier les meilleurs scénarios de priorisation devant être utilisés avec les deux opérateurs SCORING et AND sur le même ensemble d'apprentissage utilisé pour trouver les valeurs de capacité de Choquet. Comme pour les résultats obtenus dans la phase d'analyse des indices d'importance, nous avons également constaté que le meilleur scénario est celui donnant une priorité au critère "centres d'intérêt" des utilisateurs. Cependant, les opérateurs d'agrégation ne sont pas en mesure de quantifier le degré d'importance des critères comme c'est le cas pour l'intégrale de Choquet.

Afin de montrer l'efficacité de l'approche de personnalisation, nous comparons notre opérateur d'agrégation personnalisé Choquet, notée CHOPER *versus* l'opérateur d'agrégation Choquet classique non personnalisé. Les capacités utilisées avec l'opérateur de Choquet classique sont obtenus en appliquant l'algorithme 1 une seule fois (et non pas pour chaque utilisateur), donnant ainsi en sortie des valeurs d'importance sur les critères indépendamment des préférences individuelles de chaque utilisateur. Ceci donne lieu à une valeur de $0,86$ pour le critère centre d'intérêt et une valeur de l'ordre de $0,14$ pour le critère géolocalisation. Les mesures de précision obtenues sont moyennées sur toutes les séries de tests et pour l'ensemble des requêtes de test.

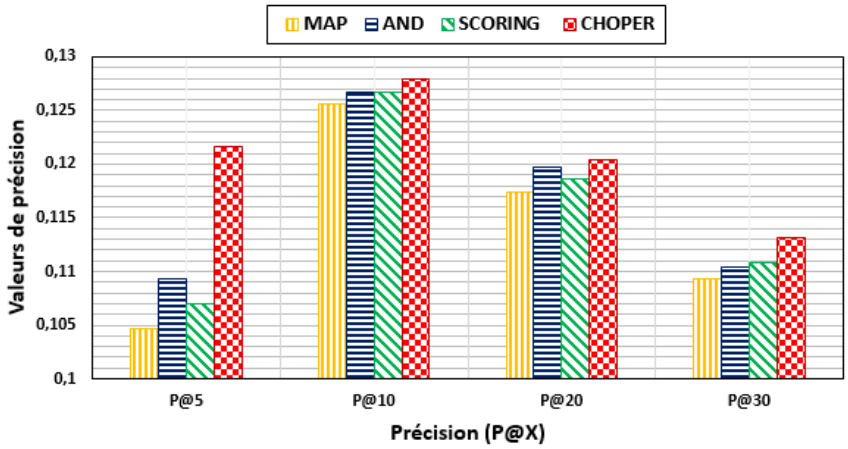


Figure 4 – Efficacité de notre approche d'agrégation de pertinence dans la tâche "Contextual Suggestion" de TREC 2013 en comparaison avec les méthodes de référence.

La figure 4 présente les résultats obtenus par notre approche CHOPER, en comparaison avec les méthodes de référence. La figure 4 montre que les performances de l'opérateur CHOPER sont significativement plus élevées que toutes les autres méthodes suivant la mesure officielle $P@5$, mais égaement suivant les autres mesures. Pour tester l'importance des améliorations obtenues par notre approche, nous avons effectué un $t-$test, et nous avons trouvé que toutes ces améliorations sont statistiquement importantes avec des $p-$valeurs < 0.01 pour toutes les fonctions d'agrégations testées.

La meilleure amélioration obtenue par notre approche suivant $P@5$ est marquée avec la méthode WAM (13.98%). En comparaison avec la meilleure méthode de référence (*i.e.,* AND), les améliorations sont significatives mais moins importantes (10.11%) en termes de $P@5$. Ces résultats sont probablement dus au fait que l'opérateur d'agrégation prioritaire AND est principalement basé sur l'opérateur MIN, ceci pourrait pénaliser les lieux pertinents selon le critère le moins important à savoir, le critère géolocalisation. Vu que la plupart des utilisateurs ont une préférence moins importante selon ce critère, la pénalisation de ce dernier permet d'améliorer les performances de recherche. La différence obtenue dans la performance, en faveur de CHOPER, s'explique par la prise en compte des différents niveaux de préférence suivant les deux critères de pertinence ainsi que la prise en compte de l'interaction qui existe entre ces derniers.

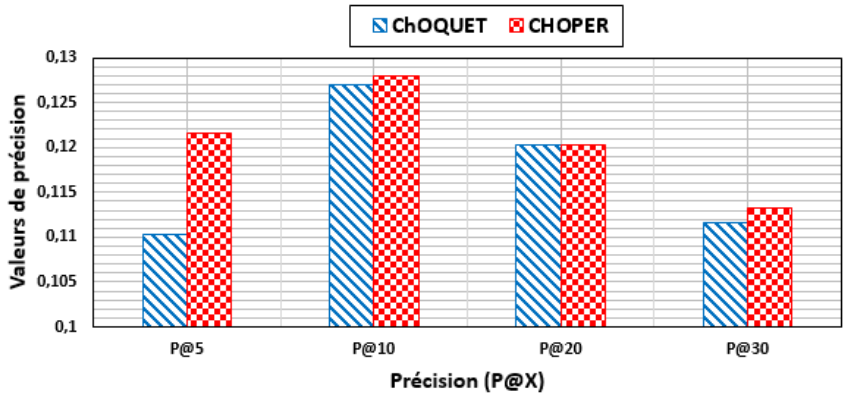


Figure 5 – Efficacité de notre approche en terme de personnalisation en comparison avec l'opérateur d'agrégation de Choquet classique.

En termes de personnalisation, la figure 5 présente les résultats obtenus en termes de précisions ($P@5$, $P@10$, $P@20$ et $P@30$) entre l'opérateur classique Choquet et sa version personnalisée CHOPER. Ces résultats montrent que le dernier est plus performant sur toutes les mesures de précision. La meilleure amélioration est de l'ordre de $9,29\%$ en termes de $P@5$. Ces résultats confirment ceux obtenus dans la phase d'identification des capacités (Cf. section 4.1) où nous avons montré que les degrés d'importance des critères dépend des préférences de l'utilisateur et ne sont pas les

mêmes pour tous. La prise en compte des poids d'importance appropriés pour chaque critère et chaque utilisateur permet de donner ainsi des résultats à la fois pertinents et adaptés aux préférences personnelles des utilisateurs.

5. Conclusion et perspectives

Dans ce papier, nous avons présenté une nouvelle approche pour l'agrégation de pertinence multidimensionnelle en tenant compte des préférences des utilisateurs. Notre approche repose sur une méthode d'agrégation floue permettant de pondérer les préférences des utilisateurs à chacun des critères agrégés. En se basant sur les indices d'importance et d'interaction, notre modèle permet de mesurer et donc d'interpréter les poids d'importance associés avec chaque critère de pertinence. L'évaluation de notre approche dans une tâche de recherche de lieux d'attraction et sur la collection de test fournie par la tâche "Contextual Suggestion" de TREC 2013, montre l'efficacité de notre approche dans l'agrégation multicritères et l'effet positif de la personnalisation des préférences des utilisateurs sur les résultats obtenus. En perspective, nous envisageons d'étendre l'approche de personnalisation proposée vers des groupes d'utilisateurs plutôt que des utilisateurs individuels. Ceci permettrait d'apprendre les préférences à partir des utilisateurs des classes similaires, permettant ainsi de pallier au problème d'insuffisance des exemples d'apprentissage.

6. Bibliographie

Baeza-Yates R. A., Ribeiro-Neto B. A., *Modern Information Retrieval*, ACM Press / Addison-Wesley, 1999.

Borlund P., « The concept of relevance in IR », *Journal of the American Society for Information Science and Technology*, vol. 54, n° 10, p. 913-925, 2003.

Bouidghaghen O., Tamine L., Pasi G., Cabanac G., Boughanem M., da Costa Pereira C., « Prioritized Aggregation of Multiple Context Dimensions in Mobile IR », *In Proceedings of the 7th Asia conference on Information Retrieval Technology*, vol. 7097 of *AIRS'11*, Springer, Berlin, Heidelberg, p. 169-180, 2011.

Choquet G., « Theory of capacities », *Annales de l'Institut Fourier*, vol. 5, p. 131-295, 1953.

da Costa Pereira C., Dragoni M., Pasi G., « Multidimensional relevance : Prioritized aggregation in a personalized Information Retrieval setting », *Information Processing and Management*, vol. 48, n° 2, p. 340-357, 2012.

Daoud M., Huang J. X., « Modeling geographic, temporal, and proximity contexts for improving geotemporal search », *Journal of the American Society for Information Science*, vol. 64, n° 1, p. 190-212, 2013.

Daoud M., Tamine L., Boughanem M., « A Personalized Graph-Based Document Ranking Model Using a Semantic User Profile », *In Proceedings of the 18th international conference on User Modeling, Adaptation, and Personalization*, UMAP'10, Berlin, Heidelberg, p. 171-182, 2010.

Daoud M., Tamine L., Boughanem M., Chebaro B., « Learning Implicit User Interests Using Ontology and Search History for Personalization », *Proceedings of the 2007 International Conference on Web Information Systems Engineering*, WISE'07, Springer-Verlag, Berlin, Heidelberg, p. 325-336, 2007.

Dean-Hall A., Clarke C., Kamps J., Thomas P., Simone N., Voorhes E., « Overview of the trec 2013 contextual suggestion track », *Text REtrieval Conference (TREC)*, National Institute of Standards and Technology (NIST), 2013.

Eickhoff C., de Vries A. P., Collins-Thompson K., « Copulas for Information Retrieval », *In Proceedings of the 36th annual International* ACM SIGIR *Conference on Research and Development in Information Retrieval*, ACM, Dublin, Ireland, 2013.

Gerani S., Zhai C., Crestani F., « Score transformation in linear combination for multi-criteria relevance ranking », *In Proceedings of the 34th European Conference on Advances in Information Retrieval*, ECIR'12, Springer-Verlag, Berlin, Heidelberg, p. 256-267, 2012.

Göker A., Myrhaug H., « Evaluation of a mobile information system in context », *Inf. Process. Manage.*, vol. 44, n° 1, p. 39-65, 2008.

Grabisch M., « Fuzzy integral in multicriteria decision making », *Fuzzy Sets and Systems*, vol. 69, n° 3, p. 279-298, 1995.

Grabisch M., Murofushi T., Sugeno M., Kacprzyk J., *Fuzzy Measures and Integrals. Theory and Applications*, Physica Verlag, Berlin, 2000.

Mata F., Claramunt C., « GeoST : geographic, thematic and temporal information retrieval from heterogeneous web data sources », *In Proceedings of the 10th international conference on Web and wireless geographical information systems*, Springer-Verlag, Berlin, Heidelberg, p. 5-20, 2011.

Moulahi B., Tamine L., Ben Yahia S., « L'intégrale de Choquet discrète pour l'agrégation de pertinence multidimensionnelle », *CORIA*, p. 399-414, 2013.

Nagmoti R., Teredesai A., De Cock M., « Ranking Approaches for Microblog Search », *In Proceedings of the 2010* IEEE/WIC/ACM *International Conference on Web Intelligence and Intelligent Agent Technology*, vol. 01 of *WI-IAT '10*, IEEE Computer Society, Washington, DC, USA, p. 153-157, 2010.

Palacio D., Cabanac G., Sallaberry C., Hubert G., « On the evaluation of Geographic Information Retrieval systems : Evaluation framework and case study », *Int. J. Digit. Libr.*, vol. 11, n° 2, p. 91-109, 2010.

Saracevic T., « Relevance : A review of the literature and a framework for thinking on the notion in information science », *Advances in Librarianship*, Academic Press, p. 79-138, 1976.

Saracevic T., « Relevance : A review of the literature and a framework for thinking on the notion in information science. Part III : Behavior and effects of relevance », *Journal of the American Society for Information Science*, vol. 58, n° 13, p. 2126-2144, 2007.

Sieg A., Mobasher B., Burke R., « Web search personalization with ontological user profiles », *In Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, ACM, New York, NY, USA, p. 525-534, 2007.

Tamine L., Bahsoun W., « Définition d'un profil multidimensionnel de l'utilisateur : Vers une technique basée sur l'interaction entre dimensions », *CORIA*, p. 225-236, 2006.

Taylor A. R., Cool C., Belkin N. J., Amadio W. J., « Relationships between categories of relevance criteria and stage in task completion », *Information Processing and Management*, vol. 43, n° 4, p. 1071-1084, 2007.

Critères numériques et temporels pour la détection de documents vitaux dans un flux

Vincent Bouvier[*,**] — **Patrice Bellot**[**]

[*] *Kware, Aix-en-Provence, France;*
[**] *Aix-Marseille Université, CNRS, LSIS UMR 7296, Marseille, France;*

RÉSUMÉ. *Cet article s'intéresse, au travers de la construction de critères numériques et temporels, à la problématique de classification de documents dans un processus de filtrage de documents vitaux. La classification a pour but de différencier les documents "vitaux" des autres documents. Un document est défini comme vital s'il concerne l'entité pour laquelle ce dernier a été sélectionné et surtout s'il contient une importante nouveauté sur une entité. Nous présentons les différents critères que nous mettons en place, dans le but de créer un modèle adaptatif qui ne dépend pas des entités sur lesquels le modèle est entraîné. La méthode est donc semi-supervisée. Ensuite, nous évaluons les résultats obtenus et nous les confrontons aux résultats de la campagne d'évaluation TREC KBA 2013 (Knowledge Base Acceleration).*

ABSTRACT. *This paper addresses to a classification challenge in a filtering task. We use different kind of features to depict vital documents and filter them out from the stream. A vital document has to be relevant for a particular entity and has to relate a new story about it. We introduce different features that uses time as well as entity profil to perform classification. We evaluate our method on framework from TREC KBA 2013 (Knowledge Base Acceleration).*

MOTS-CLÉS : *Filtrage, modèle adaptatif, profil d'entité, Random Forest,*

KEYWORDS: *Filtering, Adaptive Model, Entity Profil, Random Forest*

1. Introduction

Il existe aujourd'hui plusieurs alternatives pour suivre des informations sur le Web. Suivant la nature de l'information recherchée, il est possible d'utiliser des sites spécialisés dans l'actualité, dans les réseaux sociaux et microblogues, dans les blogues ou forums. Parcourir toutes ces sources et sélectionner les informations pertinentes est un processus long et fastidieux. Pourtant, ces informations constituent un réel besoin dans certains domaines d'applications comme la veille technologique ou encore le suivi informationnel (revue de presse). Par ailleurs, il a été observé dans l'étude de (Frank *et al.*, 2012), que les bases de données collaboratives, comme Wikipédia, souffrent de leurs grandes envergures qui ne permet pas un maintien à jour toujours efficace des informations qu'elles contiennent. Cette étude montre qu'il existe un temps médian de 356 jours entre le moment où une information concernant un sujet ou une entité relativement peu populaire apparait sur le Web et le moment où elle est relayée sur Wikipédia.

Ces problématiques soulèvent un réel besoin de veille informationnelle, qui permet de détecter de nouvelles informations sur un sujet en particulier à partir d'un certain nombre de sources. Le fait de traiter, de manière séquentielle, les documents provenant d'une ou plusieurs sources afin d'extraire ceux qui sont pertinents par rapport à un sujet correspond à la notion de *filtrage de documents* (Belkin et Croft, 1992). À la différence d'un système de RI classique, s'appuyant sur un index, le filtrage traite à la volée les documents dès leurs apparitions sur ce que l'on appelle "un flux".

Il est possible de définir un sujet à l'aide de simples mots clés, de phrases ou encore d'un ensemble de documents. Nous proposons de nous intéresser au processus de filtrage de documents lié à une entité en passant par la construction de profils d'entités. Un profil d'entité a pour rôle de représenter une entité et ses caractéristiques intrinsèques au travers de structures comme des modèles de langue, des graphiques de relation entre entités... Nous utilisons ces profils pour calculer les valeurs associées aux critères numériques et temporels pour chacun des documents filtrés, afin de définir leurs degrés de pertinence à l'aide d'une méthode de classification.

La méthode proposée a la particularité d'être indépendante des entités sur lesquelles le modèle de classification est créé. Il est alors possible d'utiliser ce modèle pour classer les documents qui traitent d'entités non présentes lors de la phase d'entrainement.

Pour tester notre méthode, nous avons utilisé l'environnement fourni pour la tâche KBA 2013 (Knowledge Base Acceleration) de la campagne d'évaluation TREC (Text RETrieval Conference).

La suite de cet article se compose d'une partie qui décrit la tâche KBA 2013 afin de prendre connaissance des particularités de la tâche et des différentes classes de documents que l'on devra filtrer. Ensuite, nous étudierons, dans une partie état de l'art, des méthodes de filtrages et de classification de documents selon des critères particuliers. Nous détaillerons les différents critères mis en place et nous étudierons

leurs impacts sur la classification de documents. Enfin, nous comparerons nos résultats à ceux soumis par les participants de la tâche KBA 2013 à TREC. Finalement, nous donnerons nos conclusions et perspectives.

2. Description de la tâche KBA 2013

La tâche KBA est directement liée au problème de mise à jour de bases de connaissances énoncé précédemment. Cette tâche a été lancée pour la première fois en 2012, où un simple filtrage de documents était demandé aux participants sans inclure de notion de nouveauté. Cependant, il ne s'agit pas d'une nouvelle tâche ad hoc de RI. En effet, l'originalité réside en partie dans le corpus appelé *"stream-corpus"* que l'on traduira par flux de documents.

Ce corpus contient environ un milliard de documents (en 2013) issus du Web et, plus spécifiquement de site d'actualité, des forums ou encore des blogues. Le point commun entre toutes ces catégories de site réside dans le fait que les documents sont datés. Les participants doivent trouver et filtrer des documents qui concernent un ensemble d'entités (sélectionnées par les organisateurs) en parcourant le corpus de manière chronologique. Il est également demandé de donner une classe aux documents filtrés parmi les quatre classes suivantes : vide/inutile, neutre, utile et vitale. Nous nous intéresserons en particulier aux classes "utile" et "vitale" qui respectivement définissent un document contenant une information connue et un document contenant une nouvelle information. La différence entre ces deux classes est très mince, ce qui ne rend pas la tâche de différenciation triviale. La prise de décision concernant la classe d'un document doit être immédiate et donc il n'est pas possible de prendre de décision a posteriori en observant des documents apparaissant plus tard dans la chronologie.

3. État de l'art

Le meilleur système de KBA 2012 (Kjersten et McNamee, 2013) utilise un système de classification simplement en créant un modèle par entité pour catégoriser les documents. Ce système impose d'avoir des données d'entrainement pour chaque nouvelle entité que l'on souhaite suivre. Cette contrainte est très forte et donc il faudra trouver un moyen de généraliser la classification de documents afin que le modèle puisse s'appliquer à n'importe quelles entités.

Nous avons, au travers d'une précédente étude (Bonnefoy *et al.*, 2013a ; Bonnefoy *et al.*, 2013b), réalisé des travaux relativement proches de ceux de (Zhou et Chang, 2013) dans le sens où nous avons essayé de généraliser le problème plutôt que d'y répondre spécifiquement pour l'ensemble d'entités fournies. Nous avons utilisé l'algorithme de classification Random Forest pour tenter de corréler un ensemble de valeurs de caractéristiques (équivalentes aux méta caractéristiques) afin de déterminer la pertinence d'un document. Il a été montré dans une étude de (Huang *et al.*, 2003), que les algorithmes de classification Naive Bayes, Random Forest ou SVM offrent

des performances proches sur plusieurs corpus. Nous avons décidé d'utiliser le Random Forest qui permet, à l'aide des Variables d'Importances, d'étudier les critères prédominant dans la classification (Breiman, 2001). Nous étions donc en mesure de catégoriser n'importe quelles entités sans pour autant avoir un corpus d'entrainement pour celles-ci. Les limites de cette méthode résident dans le choix des caractéristiques qui ne permettaient pas toujours de répondre au problème de manière efficace. Nous avons également tenté d'ajouter des caractéristiques liées à la temporalité, mais celles-ci dégradaient (en partie) les résultats.

Le filtrage de documents concernant une entité est une tâche difficile. Pour cela, il faut déjà que l'entité soit *"bien définie"* de manière à permettre une détection sur le flux de documents. Nous nous sommes alors intéressés à la construction de profils d'entités. (Li *et al.*, 2003) proposent une méthode pour extraire des profils d'entités en utilisant des patrons d'extraction. Pour cela, ils définissent la notion de profil d'entité comme étant une matrice d'attributs-valeurs en ne prenant pas en compte les possibles relations entres attributs, ni le contexte dans lequel l'entité apparait.

L'article de (Sehgal et Srinivasan, 2007) montre la construction de profil en utilisant, des documents résultant de requêtes émises sur l'API "Google Search". Il relève ensuite la similarité du profil avec la page Wikipédia de l'entité recherchée pour évaluer sa méthode. Il obtient des similarités plus élevées lorsque le profil est construit en utilisant l'intégralité des tops n documents trouvés. Cet article montre qu'il est possible d'obtenir une représentation d'une entité à partir de documents issus du Web, en utilisant une page Wikipédia comme support de comparaison. En considérant le problème à l'inverse, on peut faire l'hypothèse qu'une page issue de Wikipédia et focalisée sur une entité peut permettre d'avoir une représentation globale de ce qu'est l'entité. À minima, cela permet d'avoir une connaissance sur le contexte dans lequel l'entité est susceptible d'apparaitre.

En 2013, sur la tâche KBA, certains systèmes se sont fortement inspirés des meilleurs modèles de l'année précédente (Bellogín et Gebremeskel, 2014 ; Wang *et al.*, 2014). En revanche, la méthode de (Efron, 2014) montre un nouvel aspect dans les profils : le dynamisme. Jusqu'alors les profils qui étaient construits n'avaient pas pour vocation d'évoluer. Un profil est constitué d'un modèle de langue auquel sont ajoutés les mots présents dans les documents jugés pertinents. Cette méthode constitue une première étape dans la réalisation de notre but, mais elle est trop arbitraire. En effet, en cas d'erreur sur le jugement du document, tout le modèle de langue est altéré. Par ailleurs, pour éviter de faire grossir le modèle de langue trop fortement, ils limitent le nombre de documents qui permettent de constituer le modèle en créant une fenêtre glissante sur les documents jugés pertinents.

4. Critères de classification pour le filtrage

Un des principaux challenges consiste à trouver un ensemble de critères qui permet de caractériser au mieux la notion de document utile et vital sans tenir compte de

l'entité qui est concernée. Il sera alors tout à fait possible de calculer ces critères pour une entité pour laquelle aucune donnée d'entrainement n'a été fournie. Il y aura donc qu'un seul modèle.

Pour avoir une idée des critères pouvant être pertinents pour la détection de documents vitaux, nous avons tout d'abord procédé à une analyse des documents disponibles dans le corpus d'entrainement de KBA. Nous avons distingué ainsi trois types de critères que nous allons détailler dans les sous-parties suivantes :

- Les critères de correspondance avec l'entité ;
- Les critères du document ;
- Les critères temporels.

4.1. *Les critères de correspondance avec l'entité*

Une entité ou un sujet peut être décrit de plusieurs manières. Nous parlerons à présent de la notion de profil d'entité comme étant une structure qui permet de rassembler des informations sur une entité. Dans le cadre de la tâche KBA, 150 entités ont été sélectionnées par les organisateurs. Nous ferons référence à ces entités en parlant des entités "Twitter" et des entités "Wikipedia" suivant leur provenance. Par ailleurs, les participants ont le droit d'utiliser une version d'une base Wikipédia antérieure au début du flux de documents (début 2012). Cependant, pour certaines entités (ex. des utilisateurs Twitter), aucune page d'information n'est présente.

Nous présentons dans la suite de cette section les différents composants du profil.

4.1.1. *Les variantes de nom*

Une entité peut être mentionnée dans un document sous différents noms que nous appelons variantes. Par exemple, Elvis Presley était souvent appelé *"le King"*. Les variantes de noms permettent le plus souvent d'augmenter le rappel des documents retrouvés.

Pour obtenir ces variantes de manière non supervisée, les observations de (Cucerzan, 2007) proposent, pour une entité pour laquelle la page Wikipédia est connue, de sélectionner à la fois tous les mots en gras dans le premier paragraphe de la page ainsi que toutes les légendes des liens qui pointent vers la page Wikipédia correspondant à l'entité.

Pour les entités pour lesquelles aucune page Wikipédia ne peut être associée, il peut exister d'autres alternatives selon le type d'entité. Par exemple, pour des entités issues des réseaux sociaux, il est possible d'utiliser l'identifiant de l'entité et/ou les valeurs des champs "nom" et "prénom" lorsque ces derniers sont indiqués. Sur Twitter par exemple, l'identifiant correspond au nom d'utilisateur commençant par `@`. Par ailleurs, il est aussi possible de connaitre le nom et prénom de la personne directement sur la page du profil de l'utilisateur en question.

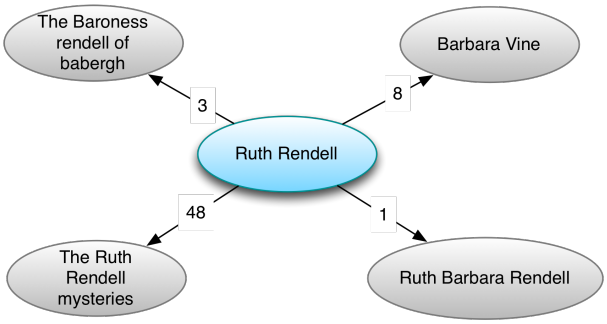


Figure 1. *Graphe des variantes pour l'entité cible Ruth Rendell avec sur chaque lien un poids qui correspond au nombre de fois où la variante a été trouvée.*

Le calcul des différents critères sur un document est un processus couteux en terme de temps de calcul. C'est pourquoi, nous proposons d'effectuer un premier filtrage de documents en se basant sur les mentions des variantes de noms à l'intérieur des documents. La manière la plus naïve et permissive (dans une optique d'obtenir un rappel plus important) consiste à considérer tous les documents ayant au moins une mention d'une entité. Les documents ainsi filtrés sont alors soumis au processus de classification.

Dans le processus de classification, différents critères sont calculés. Nous proposons d'utiliser les variantes de noms comme un critère en utilisant la mesure *TF* (Term Frequency). Le *TF.IDF* est une mesure souvent utilisée en RI. Cette mesure permet de calculer l'importance d'un mot dans un document (TF) en tenant compte de la rareté de ce mot dans un corpus de documents (IDF : Inverse Document Frequency). Le calcul de l'IDF est problématique sur un flux de documents, car ce dernier change au fur et à mesure que le flux de documents se déroule. Nous proposons dans un premier temps de n'utiliser que le TF calculé pour une entité e ayant un ensemble de variantes de noms V_e où toutes les mentions des variantes $v \in V_e$ trouvées par la fonction $f(v, D)$ sont sommées puis normalisés par la taille $|D|$ d'un document D (equation 1).

$$tf(V_e, D) = \frac{\sum_{v \in V_e} f(v, D)}{|D|} \quad [1]$$

Les critères qui utilisent les variantes de noms sont répertoriés dans le tableau 1. Les variantes de noms sont également utilisées pour construire un snippet (extrait) du document. Il est possible de construire un tel extrait en agglomérant tous les paragraphes dans lesquels il existe une mention de l'entité (voir figure 2).

tf_title	# mentions dans le titre
$tf\_document$	# mentions dans le document

Tableau 1. *Critères liés aux mentions dans le document*

Figure 2. *Snippet (extrait) de documents contenant des mentions de l'entité Edgar M. Bronfman*

4.1.2. *Le modèle de langue de l'entité*

Nous souhaitons mettre en place des critères qui permettent de mesurer la similarité d'un document avec un profil. Pour cela, un modèle de langue par entité doit être construit.

Pour les entités "Wikipédia", le modèle sera constitué à l'aide de la page associée à l'entité. Concernant les entités Twitter, le modèle sera vide. En effet, il serait tout à fait possible d'utiliser la description présente sur le profil, mais pour des raisons de cohérence dans le temps, nous ne pouvons pas les utiliser pour notre expérimentation.

Le modèle de langue que nous avons choisi est une représentation en sac de mots du document. Nous effectuons un pré-traitement sur les mots : nous ignorons les mots récurrents de la langue anglaise (utilisation d'une liste) ; les mots sont lemmatisés et mis en minuscule.

La similarité Cosine (équation 2) peut être utilisée directement sur le vecteur représenté par le sac de mots. Étant donnés deux vecteurs de mots θ et D représentant respectivement le modèle de langue d'une entité et le document, il est possible de

calculer la similarité cosinus en utilisant le produit scalaire de θ et D divisé par le produit des normes.

$$cos(\theta, D) = \frac{\theta \cdot D}{||\theta|| \cdot ||D||} = \frac{\sum_{i=1}^{n} \theta_i \times D_i}{\sqrt{\sum_{i=1}^{n} (\theta_i)^2} \times \sqrt{\sum_{i=1}^{n} (D_i)^2}} \quad [2]$$

La représentation sac de mots permet également d'utiliser des mesures telles que la divergence $js(P_\theta, P_D)$ de Jensen-Shannon (Endres et Schindelin, 2003) à condition d'utiliser des distributions de probabilités P_θ et P_D pour représenter respectivement le modèle de langue de l'entité θ et le document D.

$cos(\theta_e, D)$	similarité : modèle θ_e vs document D
$cos(\theta_e, D_{snippet})$	similarité : modèle θ_e vs snippet
$js(P_{\theta e}, P_D)$	divergence : modèle θ_e vs document
$js(P_{\theta e}, P_{D\,snippet})$	divergence : modèle θ_e vs snippet
$size(\theta_e)$	taille du modèle θ_e

Tableau 2. *Critères liés au modèle de langue θ_e d'une entité e*

4.1.3. *Les liens entre entités*

Il est possible de représenter les bases de connaissances, ou encore les réseaux sociaux comme un ensemble de graphes orientés où les noeuds correspondent à une information et les liens correspondent aux liens qui existent entre les informations. L'information dans le cas de Wikipédia correspond à une page. Dans le cas d'un réseau social, l'information concerne le plus souvent l'utilisateur. Il est possible de distinguer plusieurs types de liens par rapport à leurs orientations :

liens entrants : correspond au lien qui arrive sur le noeud ;

liens sortants : correspond au lien qui part vers un noeud différent ;

liens réciproques : lorsqu'il existe un lien entrant et sortant entre deux mêmes noeuds.

Pour établir les critères c_{lien} pour chacun des types de liens, nous calculons la fréquence d'apparition $tf(e, D)$ dans le document D de l'entité e liée au profil. Cette fréquence est normalisée par le nombre d'entités appartenant au type de lien $|L_{type}|$:

$$c_{lien}(L_{type}, D) = \frac{\sum_{e}^{L_{type}} tf(e, D)}{|L_{type}|} \quad [3]$$

4.2. *Les critères du document*

Il est important de prendre en compte des critères intrinsèques au document afin d'apporter une information supplémentaire sur la richesse d'information contenue dans le document indépendamment d'une entité. En effet, en utilisant l'entropie, il est possible d'avoir un indicateur sur la quantité d'information délivrée par un document.

Dans les critères présentés dans la table 1, un critère sur le nombre de mentions dans le titre est calculé. Cependant, nous souhaitons être capables de distinguer les cas où un document n'a pas de titre, des cas où un document à un titre. Pour cela, nous ajoutons un critère de type booléen.

4.3. *Les critères temporels*

Nous avons observé la distribution des documents par classes sur un axe temporel. La figure 3 montre que le nombre de documents (axe des ordonnées) peut varier très fortement d'un jour à l'autre (axe des abscisses), caractérisant ainsi le phénomène de rafale (ou "burst" en anglais). Ce phénomène peut apparaitre sur des documents utiles comme sur des documents vitaux. Ce critère peut être un bon indicateur sur la pertinence du document.

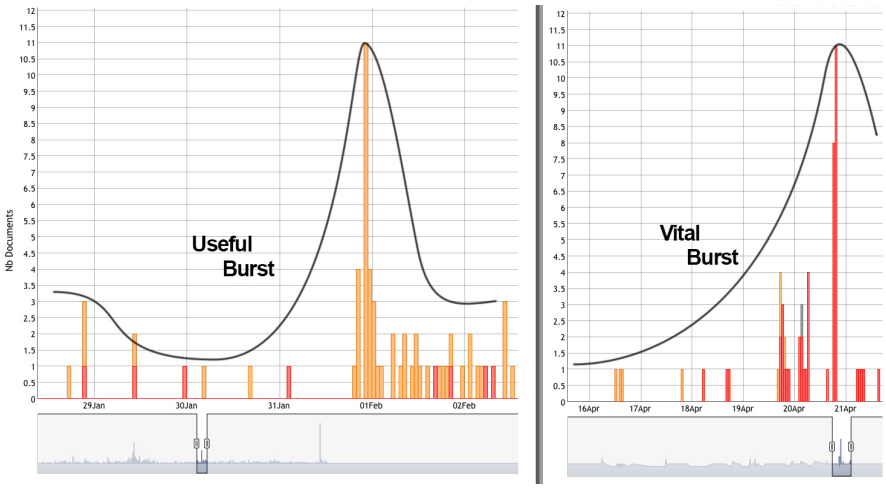


Figure 3. *Le phénomène de rafale, constaté sur deux entités différentes, ne donne pas forcément lieu à des documents dits vitaux. À gauche documents utiles, à droite documents vitaux.*

Pour caractériser l'effet rafale, nous avons utilisé une implémentation de l'algorithme de (Kleinberg, 2002) qui permet de déterminer la "force" de la rafale

d'après une analyse de série chronologique (time series). Il est possible d'utiliser différentes échelles pour calculer la série chronologique. Nous utilisons une échelle basée sur une heure pour calculer la série chronologique. L'algorithme permet également de donner la tendance de la rafale (montante ou descendante). Nous utiliserons enfin cette caractéristique directement sur le score de la force en appliquant un coefficient de -1 lorsque la rafale est descendante.

Aussi nous calculons une caractéristique basée sur le nombre de documents dans lesquels il y a une mention de l'entité les 24 heures précédant l'apparition du document évalué (voir tableau 3).

Nous avions constaté, à la suite d'expérimentations supplémentaires sur nos études (Bonnefoy *et al.*, 2013a ; Bonnefoy *et al.*, 2013b), que certains critères temporels tendaient à dégrader les résultats. Nous avons finalement choisi de ne garder que les critères en correspondance avec le nombre de mentions trouvés les dernières 24 heures et celui sur la force (et direction) d'un éventuel effet de rafale par rapport aux observations décrites précédemment.

$kleinberg_{1h}$	force et direction de la rafale
$match_{24h}$	nombre de documents trouvés les dernières 24h

Tableau 3. *Critères liés à la date d'apparition du document*

5. Expérimentations

Nous avons utilisé le corpus d'entrainement fourni par les organisateurs de TREC KBA pour entraîner des algorithmes de classification de types *"Random Forest"* sur les documents annotés. Nous parcourons le corpus de manière chronologique afin de simuler un flux de documents. Nous pouvons ainsi calculer toutes les caractéristiques précédemment présentées.

Nous avons testé quatre systèmes différents :

- le premier *"Single"* entraine un algorithme de classification qui connait les quatre classes de documents de KBA (Garbage, Neutral, Useful, Vital). La réponse de ce dernier donne la classe du document ;

- le second système *"TwoStepClassifier"* est composé de deux algorithmes de classification utilisés en cascade. Chacun s'entraîne sur 2 classes. Le premier donne une classe parmi : *"Garbage/Neutral"* et *"Useful/Vital"*. Le second algorithme de classification donne une classe parmi : *"Useful"* et *"Vital"* ;

- le troisième système *"VitalvsOthers"* entraine un algorithme de classification qui apprend à distinguer seulement la classe *"Vital"* contre toutes les autres classes (*"Others"*).

Le quatrième système *"Combine"* est quant à lui un peu différent puisqu'il va apprendre les scores issus des trois précédents systèmes afin de trouver la meilleure combinaison possible.

6. Résultats

Nous avons utilisé l'outil d'évaluation officiel pour pouvoir comparer nos résultats à ceux des autres participants de KBA 2013 sur la partie test du corpus. La mesure utilisée pour l'étude est la f-mesure ($f_1(p, r)$) qui est une moyenne harmonique entre la précision p et le rappel r (voir équation 4).

$$f_1(p, r) = \frac{2 * p * r}{p + r} \quad [4]$$

Dans l'évaluation de KBA 2013, il y a deux aspects pris en compte : la recherche d'information et la classe donnée à un document. Dans notre analyse, nous comparons nos scores obtenus avec l'outil d'évaluation officiel afin de servir de référence. Nous analyserons aussi d'autre part l'efficacité de la classification seule en prenant en compte un système de RI parfait.

6.1. *Analyse des résultats dans le cadre de KBA 13*

Nous pouvons voir d'après le tableau 4 que nous obtenons des résultats supérieurs au meilleur résultat présenté lors de la campagne d'évaluation pour la majorité de nos systèmes sur la classification de documents vitaux. De plus, nous remarquons que d'apprendre à combiner les scores des différents systèmes s'avère être une stratégie payante.

Système	f_1
Best KBA	.360
Mean KBA	.193
Single	.395
TwoStep	.388
VitalVsOther	.346
Combine	**.403**

Tableau 4. *Récapitulatif des résultats présentés à TREC KBA et de ceux de nos différents systèmes*

Les scores de KBA sont observés en utilisant plusieurs seuils suivant le score de confiance donné par l'algorithme de classification (score allant de 0 à 1000). Le

graphique 4 montre une cohérence dans le score de confiance donné à l'issue de la classification. En effet, plus le score est haut (proche de l'axe des ordonnées) plus la précision monte. En revanche, pour les autres systèmes, les pourcentages de précision, rappel et f-mesure s'écroulent lorsque le score de confiance augmente.

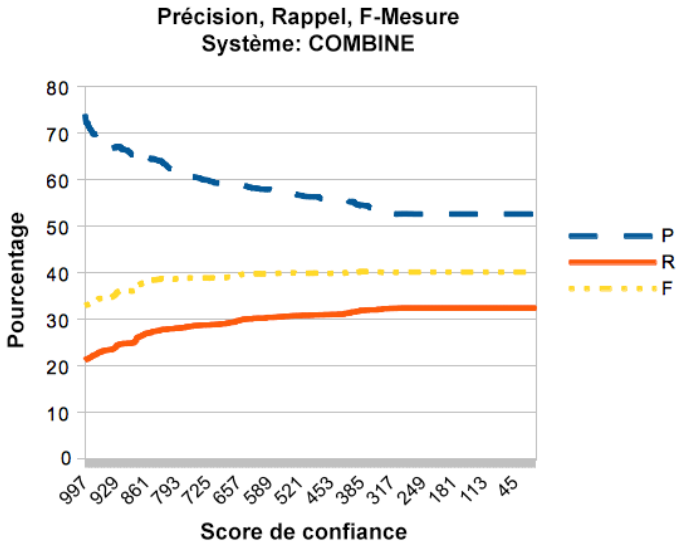


Figure 4. *Variation de la Précision (P), du Rappel (R) et de la F-mesure (F) en fonction du score de confiance du système Combine.*

Si l'on regarde d'un point de vue seulement de classification de documents (table 5), on remarque que les scores de classifications sont plutôt bons, ce qui montre une lacune de notre système sur le processus de recherche d'information. Pour cette analyse, nous utilisons une matrice de confusion (Vrais Positifs, Vrais Négatifs, Faux Positifs, Faux Négatif) pour calculer les scores de précision, rappel et f-mesure.

Système	Precision	Rappel	F-Mesure
Single	.569	.406	**.474**
TwoStep	.475	**.436**	.455
VitalVsOther	**.725**	.323	.447
Combine	.619	.368	.461

Tableau 5. *Résultats de la classification sans prendre en compte le processus de RI*

Cette analyse nous permet de voir que le système *"VitalVsOther"* offre la précision la plus haute au détriment du rappel. Nous pouvons constater également que le système *"Combine"* tente de tirer parti de tous les systèmes en offrant une précision

relativement élevée et un rappel plus élevé que *"VitalVsOther"*. C'est d'ailleurs *"Combine"* qui obtient le deuxième meilleur score en f-mesure.

Nous avons voulu en savoir plus sur les critères prédominants dans le processus de décision de la classe vital. Pour cela, nous avons utilisé le logiciel R et la bibliothèque "Party" qui implémente un algorithme de classification de type *"Random-Forest"* pour lequel il est possible de calculer les Variables d'Importances (VI). Les VI sont calculées à l'aide d'une permutation aléatoire des valeurs des différents critères pour finalement, calculer la différence de précision "avant/après" révélant ainsi l'importance du critère (Breiman, 2001).

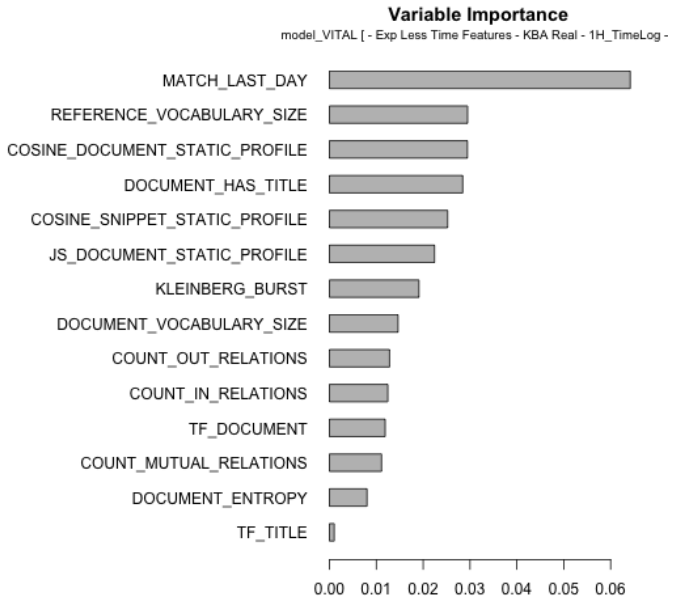


Figure 5. *Classement des critères en fonction de l'importance dans la classification, avec en abscisse l'importance relative des critères.*

La figure 5 montre que le critère le plus important dans la classification correspond au critère qui regarde le nombre de documents apparus les 24 heures précédentes et portant au moins une mention de l'entité. Cela montre qu'effectivement le temps peut jouer un rôle très important dans la manière dont est publiée une nouvelle information. Cependant, il faut tout de même être prudent sur l'utilisation de tels critères. En effet, les valeurs calculées pour certains critères peuvent être très variables en fonction des entités. Par le fait, ces critères ne sont peut être pas adaptés à un modèle généraliste.

On notera cependant que le critère mesurant la rafale de Kleinberg ne joue pas un rôle important. En observant l'allure du critère sur les modèles, les scores sont tous négatifs. Autrement dit, il ne capte que des rafales descendantes. Nous projetons de faire une étude un peu plus poussée afin de trouver comment utiliser de manière efficace les différents paramètres de l'algorithme.

Le deuxième critère correspond à la taille du modèle de langue de l'entité. Le modèle a très bien su s'adapter aux différences qu'il peut y avoir entre les entités de type Wikipédia et Twitter où, pour ces dernières, le modèle de langue est vide.

7. Conclusion et Perspectives

Nous avons présenté dans cet article trois différents types de critères permettant la classification de documents vitaux pour une entité. Nous avons également montré que le modèle généré à l'issue de l'entrainement n'est pas dépendant des entités sur lesquelles il est créé. Finalement, nous avons montré les performances de notre système en nous comparant aux résultats de la campagne d'évaluation TREC KBA 2013.

Nous planifions de revoir nos paramètres sur le critère de rafale afin de voir s'il est possible de trouver un bon réglage afin de détecter la rafale au moment de la montée, ce qui nous semble plus logique pour la découverte de documents vitaux. Nous avons vu dans l'état de l'art une étude (Efron, 2014) sur l'évolution du profil d'entité en fonction du temps. Nous aimerions nous investir dans cette voie afin de voir s'il est possible d'adapter notre système pour prendre en compte le dynamisme des entités. Enfin, nous voudrions retravailler notre méthode préliminaire de recherche d'information, afin d'améliorer le rappel de notre système. Ici, nous utilisons les variantes de nom issues de Wikipédia, cependant nous ne les mettons jamais à jour. Par exemple, il serait intéressant de prendre en considération un système d'extraction d'entités nommées afin de permettre d'extraire de nouvelles variantes de noms.

8. Bibliographie

Belkin N. J., Croft W. B., « Information filtering and information retrieval : two sides of the same coin ? », *Communications of the ACM*, vol. 35, nº 12, p. 29-38, December, 1992.

Bellogín A., Gebremeskel G., « CWI and TU Delft Notebook TREC 2013 : Contextual Suggestion, Federated Web Search, KBA, and Web Tracks », *proceedings of the Twenty-Second Text REtrieval Conference (TREC 2013) Gaithersburg, Maryland, November 19–22, 2013*, National Institute of Standards and Technology (NIST), p. 500-302, 2014.

Bonnefoy L., Bouvier V., Bellot P., « LSIS/LIA at TREC 2012 knowledge base acceleration », *Proceedings of the Twenty-First Text REtrieval Conference (TREC 2012)*, p. 500-298, 2013a.

Bonnefoy L., Bouvier V., Bellot P., « A weakly-supervised detection of entity central documents in a stream », *The 36th International ACM SIGIR conference on research and development in Information Retrieval, SIGIR '13, Dublin, Ireland - July 28 - August 01, 2013*, p. 769-772, 2013b.

Breiman L., « Random Forests », *Machine Learning*, vol. 45, n° 1, p. 5-32, 2001.

Cucerzan S., « Large-Scale Named Entity Disambiguation Based on Wikipedia Data », *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning,(EMNLP-CoNLL 2007)*, ACL, p. 708-716, 2007.

Efron M., « The University of Illinois' Graduate School of Library and Information Science at TREC 2013 », *proceedings of the Twenty-Second Text REtrieval Conference (TREC 2013) Gaithersburg, Maryland, November 19–22, 2013*, National Institute of Standards and Technology (NIST), p. 500-302, 2014.

Endres D. M., Schindelin J. E., « A new metric for probability distributions », *IEEE Transactions on Information Theory*, vol. 49, n° 7, p. 1858-1860, 2003.

Frank J., Kleiman-Weiner M., Roberts D. A., Niu F., Zhang C., « Building an Entity-Centric Stream Filtering Test Collection for TREC 2012 », *Proceedings of the Twenty-First Text REtrieval Conference (TREC 2012) Gaithersburg, Maryland, November 6-9, 2012*, National Institute of Standards and Technology (NIST), p. 500-298, 2012.

Huang J., Lu J., Ling C. X., « Comparing Naive Bayes, Decision Trees, and SVM with AUC and Accuracy. », *ICDM*, IEEE Computer Society, p. 553-556, 2003.

Kjersten B., McNamee P., « The HLTCOE approach to the TREC 2012 KBA track », *proceedings of the Twenty-First Text REtrieval Conference (TREC 2012) Gaithersburg, Maryland, November 6-9, 2012*, National Institute of Standards and Technology (NIST), p. 500-298, 2013.

Kleinberg J., « Bursty and hierarchical structure in streams », *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, July 23-26, 2002, Edmonton, Alberta, Canada*, p. 91-101, 2002.

Li W., Srihari R., Niu C., « Entity profile extraction from large corpora », *Proceedings of Pacific Association for Computational Linguistics (PACLING 2003)*, 2003.

Sehgal A. K., Srinivasan P., « Profiling Topics on the Web », *Proceedings of the WWW2007 Workshop I^3 : Identity, Identifiers, Identification, Entity-Centric Approaches to Information and Knowledge Management on the Web, Banff, Canada, May 8, 2007*, p. 1-8, 2007.

Wang J., Song D., Lin C.-Y., Liao L., « BIT and MSRA at TREC KBA CCR Track 2013 », *proceedings of the Twenty-Second Text REtrieval Conference (TREC 2013) Gaithersburg, Maryland, November 19–22, 2013*, National Institute of Standards and Technology (NIST), p. 500-302, 2014.

Zhou M., Chang K. C.-C., « Entity-centric document filtering : boosting feature mapping through meta-features », *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, ACM, p. 119-128, 2013.

Session 6

Services

DaWeS: Data Warehouse fed with Web Services

John Samuel, Christophe Rey

LIMOS, CNRS
Université Blaise Pascal
Aubière, France
samuel@isima.fr, christophe.rey@univ-bpclermont.fr

RÉSUMÉ. *Nous présentons un prototype, appelé DaWeS, d'entrepôt de données alimenté par des services web. La spécificité de DaWeS est son approche médiation (intégration de données sans matérialisation) comme outil ETL (extraction, transformation et chargement des données). Cette approche permet d'automatiser une grande partie du processus ETL, tout en facilitant les interventions humaines par l'emploi exclusif de langages déclaratifs (requêtes datalog, SQL, XSD, XSLT). Le contexte de cette étude est celui des standards relatifs aux services web les plus utilisés car les plus simples (HTML, HTTP, REST, XML, JSON), et non des standards plus élaborés mais moins utilisés (SOAP, UDDI, WSDL, SA-WSDL, OWL-S, hRESTS). En termes applicatifs, l'ambition est de permettre à l'administrateur de DaWeS de proposer aux petites et moyennes entreprises un service de stockage et d'interrogation de leurs données métier liées à l'utilisation de services web tiers, sans avoir elles-mêmes à gérer leur propre entrepôt. En particulier, DaWeS permet la définition facile d'indicateurs de performance personnalisés.*

ABSTRACT. *We present a prototype, called DaWeS, which is a Data Warehouse fed with Web Services. The main feature of DaWeS is to use a mediation approach (data integration without materialization) as the ETL tool (data extraction, transformation and loading). This approach enables to automate many steps of the ETL process, while facilitating human interventions by exclusively relying on declarative languages (datalog queries, SQL, XSD, XSLT). The context of this work consists of the mostly used (because the simplest) web services standards (HTML, HTTP, REST, XML, JSON), and not of the more complex but less used ones (SOAP, UDDI, WSDL, SA-WSDL, OWL-S, hRESTS). In terms of applications, the aim is to allow a DaWeS administrator to provide to small and medium companies a service to store and query their business data coming from their usage of third-party services, without having to manage their own warehouse. In particular, DaWeS enables the easy design of personalized performance indicators.*

MOTS-CLÉS : *médiation, entrepôt de données, services web, ETL, intégration de données, réécriture de requêtes*

KEYWORDS: *mediation, data warehouses, web services, ETL, data integration, query rewriting*

1. Introduction

The past two decades have seen the rise of many Web Services (WS) providers offering a reduced subset of services rather than the traditional bloated software applications. These services are heterogeneous, autonomous and ever evolving. Enterprises using WS have no direct control over the underlying data infrastructure and thereby over their own business data. The only convenient mechanism for enterprises to access and manipulate their data is through application programming interface (API) exposed by service providers to allow the clients to build their own internal dashboards. WS APIs differ among each other significantly with respect to the use of different message formats, authentication mechanisms, service level agreements, access patterns, data types, and the choice of input, output and error parameters. APIs are mostly described using human readable (HTML) web pages. Furthermore service providers often make updates to their services (addition and deprecation of resources, change in the API or SLA). These changes may lead to the losing of past enterprise data. All the aforementioned challenges make it difficult for small and medium scale enterprises with lesser human resources and expertise to easily integrate with numerous WS.

Companies traditionally use a data warehouse to perform business analysis, compute performance measures (aka indicators) and track their growth. The purpose of our work is to aid enterprises using WS for their day to day business activities with a data warehouse service. We are building a multi-enterprise *Data Warehouse fed with Web Services (DaWeS)* able to fetch interesting data from various WS and expose them in a manner so that the end users can compute their own interesting performance indicators without having to manage their own warehouse.

In this paper, we present an experimental study of a prototype which aims at being a convenient and realistic semi-automated system. Indeed, our goal is not to build a fully automated WS fed data warehouse, which seems quite impossible, but to ease as much as possible the coding burden of adding and updating new WS (achievable by only a couple of developers or administrators). The feeding of the data warehouse is achieved through the use of mediation techniques associated to a generic wrapper. Though our experiments focus on REST (like) services given their popularity, it can be easily extended to SOAP services and the possible future availability of machine readable standards like WSDL will further reduce the administrators' tasks.

In section 2, we concretely explore three domains to determine what are the mostly used WS standards. Section 3 formally presents the mediation approach, coupled with the generic wrapper, used to feed the data warehouse. Section 4 describes DaWeS architecture, development and the various experiments. Section 6 describes the various related works. Finally we conclude by describing current and future works.

2. Data Warehouse and Web Services

This section surveys 12 WS belonging to three business domains to establish the mostly used WS standards that are effectively used by service providers. The three

studied domains are email marketing, project management and helpdesk (support). Email marketing is a form of direct marketing which uses email campaigns as a means for communicating to a wide (subscribed) audience about new products and technologies. Project management encompasses many activities : planning and estimation of projects, decomposing them to several tasks and tracking their progress. Helpdesk is focused on managing customers' (intended or current) problems, complaints and suggestions on an online web portal internally tracked using tickets.

The 12 surveyed WS are project management (Basecamp : www.basecamp.com, Liquid Planner : www.liquidplanner.com, Teamwork : www.teamworkpm.net and Zoho Projects : www.zoho.com/projects), email marketing (iContact : www.icontact.com, CampaignMonitor : www.campaignmonitor.com and MailChimp : mailchimp.com) and helpdesk (Zendesk : www.zendesk.com, Desk : www.desk.com , Zoho Support : www.zoho.com/support, Uservoice : www.uservoice.com and FreshDesk : www.fresh desk.com). Each of the previous service may propose many operations, each of which has a callable API. The characteristics of these APIs are given in table 2 where WS are classified according to : the language in which APIs are described (i.e., documented), their REST compliance (Fielding, 2000), their version of the API, their authentication method, the resource they deal with (e.g. *task* or *todo* in a project management service, *ticket* in an helpdesk service), their message format, the used service level agreement (constraints on the operations usage) their HTTP access method, the used data types, their handling of dynamic resources (resources which value can evolve), mandatory constraints during operation invocation (e.g. to get all the tasks, it first requires in Teamwork to get all the projects, following retrieving all the task lists in all the projects and finally followed by obtaining the tasks from all the task lists), and their pagination features (i.e., one or many call(s) to retrieve all data).

From these characteristics, an average profile of WS emerges : describing services with HTML, following the REST architecture, using basic HTTP authentication with a GET access, XML or JSON as message format, enumeration and date as data types, dynamic resources and sequence operation invocation. This average profile clearly focuses on simplicity. The consequence is a low level of service management automation. For example, none of these services are described using a computer-oriented language (with or without semantic features) like WSDL (W3C, 2001), SA-WSDL (Kopecký *et al.*, 2007), DAML-S (Burstein *et al.*, 2002), OWL-S (Martin *et al.*, 2007), hRESTS (Kopecký *et al.*, 2008). This situation is also confirmed by ProgrammableWeb (ProgrammableWeb, 2013), a directory which documentes 10,555 APIs and in which a vast majority (around 69%) are REST based WS.

So the existing standards aiming at a better automation of WS management are not really used and widely spread yet. It thus seems important to investigate a semi-automated approach to build a WS fed data warehouse, keeping the requirement of reducing the code burden needed to maintain such a system.

Unlike the traditional WS discovery issue, in the DaWeS context, the services discovery does not really need to be automated since it's up to the user to inform the system about the services he's using. So, the real automation problem resides in the

Tableau 1. *Web Service API Analysis on three domains*

Project Management	**Basecamp**	**LiquidPlanner**	**Teamwork**	**Zoho Projects**
1. API Description	HTML page	HTML page	HTML page	HTML page
2. Conform to REST	REST like	REST like	REST like	Not REST
3. Version	v1	3.0.0	N.A.	N.A.
4. Authentication	Basic HTTP, OAuth 2	Basic HTTP	Basic HTTP	Basic HTTP
5. Resources Involved	Project,Todo List, Todo	Project, Task	Project, Task List, Task	Project, Task List, Task
6. Message Formats	JSON	JSON	XML, JSON	XML, JSON
7. Service Level Agreement	Max 500 requests /10s from same IP address for same account	Max 30 requests /15s for same account	Max 120 requests /1min	Error code :6403 on exceeding the limit
8. HTTP Access	GET	GET	GET	POST
9. Data Types (dt)	Enumerated dt (Project and Todo Status), Date	Enumerated dt (Project and Task Status), Date	Enumerated dt (Project and Task Status), Date	Enumerated dt (Project and Task Status), Date
10. Dynamic nature of the resources	Yes (Project and Task Status)	Yes (Project and Task Status)	Yes (Project and Task Status)	Yes(Project and Task Status)
11. Operation Invocation	Sequence Required	Sequence Not Required	Sequence Required	Sequence Required
12. Pagination	No	No	No	Yes

Email Marketing	**Mailchimp**	**CampaignMonitor**	**iContact**
1. API Description	HTML page	HTML page	HTML page
2. Conform to REST	Not REST	REST like	REST like
3. Version	1.3	v3	2.2
4. Authentication	Basic HTTP	Basic HTTP, OAuth 2	Basic HTTP (with Sandbox)
5. Resources Involved	Campaign, Campaign Statistics	Campaign, Campaign Statistics	Campaign, Campaign Statistics
6. Message Formats	XML, JSON, PHP, Lolcode	XML, JSON	XML, JSON
7. Service Level Agreement	N.A.	N.A.	75,000 requests /24h, with a max of 10,000 requests /1h
8. HTTP Access	GET	GET	GET
9. Data Types (dt)	Enumerated Data types (Campaign Status), Date	Enumerated Data types (Campaign Status), Date	Enumerated Data types (Campaign Status), Date
10. Dynamic nature of the resources	Yes (Campaign Status)	Yes (Campaign Status)	Yes (Campaign Status)
11. Operation Invocation	Sequence Required	Sequence Required	Sequence Not Required
12. Pagination	Yes	No	No

Support	**Zendesk**	**Desk**	**Zoho Support**	**Uservoice**	**Freshdesk**
1. API Description	HTML page	HTML page	HTML page	HTML page	HTML page
2. Conform to REST	REST like	REST	Not REST	REST like	REST like
3. Version	v1	v2	N.A.	v1	N.A.
4. Authentication	Basic HTTP	Basic HTTP, OAuth 1.0a	Basic HTTP	OAuth 1.0	Basic HTTP
5. Resources Involved	Forum, Topic, Ticket	Case	Task	Forum, Topic, Ticket	Forum, Topic, Ticket
6. Message Formats	XML, JSON	JSON	XML, JSON	XML, JSON	JSON
7. Service Level Agreement	Limit exists (but unknown)	60 requests per minute	250 calls /day /org (Free)	N.A.	N.A.
8. HTTP Access	GET	GET	GET	GET	GET
9. Data Types (dt)	Enumerated dt (Ticket Status), Date	Enumerated dt (Case Status), Date	Enumerated dt (Task Status), Date	Enumerated dt (Ticket Status), Date	Enumerated dt (Ticket Status), Date
10. Dynamic resources	Yes (Ticket Status)	Yes (Case Status)	Yes (Task Status)	Yes (Ticket Status)	Yes (Ticket Status)
11. Operation Invocation	Sequence Required	Sequence Required	Sequence Required	Sequence Required	Sequence Required
12. Pagination	Yes	Yes	Yes	Yes	No

automated connection between the warehouse and the known web services that will be its data sources. WSDL is typically a technology that is meant for enabling the automated generation of a wrapper between the system and a WS. But even if we could use such standard, it wouldn't solve the entire problem. Indeed, having a wrapper allows to call the WS. But it does not link the semantics of what the service can do (e.g. what kind of data it can provide) to the semantics of the system which is the one the user knows (typically given by the schema of the warehouse).

The solution we describe in the next section is to manually achieve the connection between DaWeS and WS in a twofold manner : (i) dedicating the greatest part of the manual effort to establish the semantic connection between data in DaWeS and data coming from the WS, and (ii) trying to reduce the daily coding effort to deal with syntactic mismatches. (i) will be obtained via a mediation approach, and (ii) via the building of a generic wrapper and the use of declarative languages only for each manual task.

3. Mediation as ETL

In the data integration field, mediation (Wiederhold, 1992) is the main virtual approach to provide a uniform query interface to multiple heterogeneous and autonomous data sources. Every source has its own (local) schema linked via mappings to the mediated or global schema (i.e., the schema of the mediator) over which the user queries are formulated. The approach is virtual because the global schema does not contain any data. After the user has posed her query over the global schema, this query is reformulated into a set of queries such that each of them can be posed over a special source. After each source has given its results, these are merged into the mediator and presented to the user. Among the different kind of mappings between the local and the global schema (see (Chawathe *et al.*, 1994 ; Duschka et Genesereth, 1997 ; Ullman, 2000 ; Friedman *et al.*, 1999)), the Local As View (LAV) mappings are known to allow easy addition, update and removal of sources (Ullman, 2000). Indeed, adding, updating and removing a LAV mapping can be done without modifying anything else than the mapping itself. This is due to the fact that a LAV mapping is defined as a query over the global schema. Another consequence is that defining a mapping is done using a declarative query language, and not through the programming of a piece of software, which is easier, quicker and less constraining.

Thus, the mediation with LAV mappings approach fits particularly well our need to easily (thus manually) connect to multiple and heterogeneous WS. It becomes the ETL (extraction-transformation-loading) tool of our data warehouse. This implies WS are considered as relational data sources. Since the access to WS is constrained by precise input values, to get output data, then WS must be considered as relational data sources with access patterns that specify which attributes are inputs and which are outputs. More precisely, each operation provided by a WS is modeled as a relation associated to an access pattern (Ullman, 1989) whose size is equal to the number of attributes in a relation. Syntactically, the access pattern is represented by an adornment

Tableau 2. *Helpdesk WS and their operations*

Service	Operation name	Inputs	Outputs
Desk v2 API	Deskv2TotalCases (D2TC)	None	Total nb of tickets : $pgno, pgsize$
	Deskv2Case (D2C)	$pgno$, $pgsize$	One page tickets details : $tkid$, tkn, $tkcd$, tkp, tks
Zendesk v1 API	Zendeskv1Ticket (ZT)	None	All ticket id : $tkid$
	Zendeskv1SolvedTicket (ZST)	None	All closed tickets id : $tkid$
	Zendeskv1TicketDetails (ZTD)	$tkid$	One ticket details : tkn, $tkcd$, $tkdd$, $tkcmpd$, tkp, tks
Uservoice v1 API	Uservoicev1TotalTickets (UTT)	None	Total nb of tickets : $pgno, pgsize$
	Uservoicev1Ticket (UT)	pgn, pgs	One page tickets details : id, tkn, $tkcd$, tkp, tks

being a tuple of b and f letters written besides the relation name. In this tuple, b (for "bound") in ith position says the ith attribute is an input ; f (for "free") says it is an output.

Example 1. : Let us consider three WS in the helpdesk domain : Zendesk, Uservoice and Desk. They allow customers to submit their complaints. These are tracked by tickets. Every ticket has an associated priority and status. Some need immediate attention and therefore have high priority. When a ticket is created, its status is open and when resolved, its status is completed (or closed).

Here are attribute names given to ticket related information. A page is an answer of an API call. $pgno$ is a page number, $pgsize$ is a number of tickets in one page, $limit$ is a number of results in a page, $tkid$ is a ticket identifier, tkn is a ticket name, $tkcd$ is a ticket creation date, $tkdd$ is a ticket due date, $tkcmpd$ is a ticket effective completion date, tkp is a ticket priority and tks is a ticket current status. src is a WS name, and $operation$ is an operation name.

We want DaWeS to be connected to these services so that customers can get performance indicators about the handling of their complaints. Towards this purpose, each WS offers at least one operation callable through its API (see table 1). For these services to be connected to DaWeS, the global schema must contain relations that describe the domain. Here are the two relations extracted from the global schema that describe everything linked to the notion of ticket :

$\mathbf{Ticket}(tkid, src, tkname, tkcdate, tkddate, tkcmpdate, tkpriority, tkstatus)$

$\mathbf{Page}(pgno, src, operation, limit)$

Now, the following queries define the LAV mappings between each operation and the global schema (these are conjunctive queries written in the rule-style syntax) :

$\mathbf{D2TC}^{ff}(pgno, pgsize) \leftarrow \mathbf{Page}(pgno,'\text{ Desk v2 API}','\text{ Deskv2Case}', pgsize).$

$\mathbf{D2C}^{bbffffff}(pgno, pgsize, tkid, tkn, tkcd, tkp, tks) \leftarrow$
$\mathbf{Page}(pgno,'\text{ Desk v2 API}','\text{ Deskv2Case}', pgsize),$
$\mathbf{Ticket}(tkid,'\text{ Desk v2 API}', tkn, tkcd, tkdd, tkcmpd, tkp, tks).$

$\mathbf{ZT}^{f}(tkid) \leftarrow \mathbf{Ticket}(tkid,'\text{ Zendesk v1 API}', tkn, tkcd, tkdd, tkcmpd, tkp, tks).$

$\mathbf{ZST}^{f}(tkid) \leftarrow \mathbf{Ticket}(tkid,'\text{ Zendesk v1 API}', tkn, tkcd, tkdd, tkcmpd, tkp,'\text{ Closed}').$
$\mathbf{ZTD}^{bfffffff}(tkid, tkn, tkcd, tkdd, tkcmpd, tkp, tks) \leftarrow$
$\mathbf{Ticket}(tkid,'\text{ Zendesk v1 API}', tkn, tkcd, tkdd, tkcmpd, tkp, tks).$
$\mathbf{UTT}^{ff}(pgno, pgsize) \leftarrow \mathbf{Page}(pgno,'\text{ Uservoice v1 API}','\text{ Uservoicev1Ticket}', pgsize).$
$\mathbf{UT}^{bbffffff}(pgn, pgs, id, tkn, tkcd, tks, tkp) \leftarrow$
$\mathbf{Page}(pgn,'\text{ Uservoice v1 API}','\text{ Uservoicev1Ticket}', pgs),$
$\mathbf{Ticket}(id,'\text{ Uservoice v1 API}', tkn, tkcd, tkdd, tkcmpd, tkp, tks).$

Once the global schema and the LAV mappings are built, the user is able to pose her query over the global schema without dealing with the source relations intricacies. The query will then be automatically transformed by a query rewriting algorithm into a query plan, which describes the sequence of API operation calls (from potentially different WS) needed to answer the user query.

The classical query rewriting algorithms include bucket algorithm (Levy *et al.*, 1996), inverse rules algorithm (Duschka et Genesereth, 1997 ; Duschka *et al.*, 2000) and minicon algorithm (Pottinger et Levy, 2000). They compute the so-called maximally contained rewritings which allow to obtain the certain answers of a query. Informally, this means that the computed answers are not false, at the computation time, in every source. So, for example, if two services deliver different status for the same ticket id, these status does not belong to the certain answers and will not be presented in the query result. In DaWeS, we chose inverse rules algorithm (Duschka et Genesereth, 1997 ; Duschka *et al.*, 2000) since it can handle access patterns in the data sources description and is the only algorithm (up to our knowledge) being able to rewrite datalog recursive queries posed to the global schema (for conjunctive queries as LAV mappings). Moreover it is shown in (Abiteboul et Duschka, 1998) that generating a query plan can be done in polynomial time with respect to the data complexity (i.e., in the sizes of the query and the mappings). This ensures, at least theoretically, quite good performances. Besides, various integrity constraints on the global schema like full and functional dependencies can also be handled by this algorithm.

Example 2. In the context of example 1, we now consider a record definition. Note that we use a special function here called *yesterday()*, which is executed before the query evaluation, to obtain yesterday's date. The record we define is called Daily New Tickets (DNT) : it is the number of tickets that were created yesterday.
$\mathbf{DNT}(tkid, src, tkn, tkp, tks) \leftarrow$
$\mathbf{Ticket}(tkid, src, tkn,' yesterday()', tkdd, tkcmpd, tkp, tks).$
The following program is the query plan which is the rewriting of query DNT.
$\mathbf{Page}(pgno,'\text{ Desk v2 API}','\text{ Deskv2Case}', pgsize) \leftarrow \mathbf{D2TC}^{ff}(pgno, pgsize).$
$\mathbf{DPgNo}(pgno) \leftarrow \mathbf{D2TC}^{ff}(pgno, pgsize).$
$\mathbf{DPgSize}(pgsize) \leftarrow \mathbf{D2TC}^{ff}(pgno, pgsize).$
$\mathbf{Page}(pgno,'\text{ Desk v2 API}','\text{ Deskv2Case}', pgsize) \leftarrow$
$\mathbf{DPgNo}(pgno), \mathbf{DPgNo}(pgsize), \mathbf{D2C}^{bbffffff}(pgno, pgsize, tkid, tkn, tkcd, tkp, tks)$
$\mathbf{Ticket}(tkid,'\text{ Desk v2 API}', tkn, tkcd, f_{D2C,5}(pgno, pgsize, tkid, tkn, tkcd, tkp, tks),$
$f_{D2C,6}(pgno, pgsize, tkid, tkn, tkcd, tkp, tks), tkp, tks) \leftarrow$

$\mathbf{DPgNo}(pgno), \mathbf{DPgSize}(pgsize), \mathbf{D2C}^{bbffffff}(pgno, pgsize, tkid, tkn, tkcd, tkp, tks)$
$\mathbf{Ticket}(tkid,'\text{ Zendesk v1 API}', f_{ZT,3}(tkid), f_{ZT,4}(tkid), f_{ZT,5}(tkid),$
$f_{ZT,6}(tkid), f_{ZT,7}(tkid), f_{ZT,8}(tkid)) \leftarrow \mathbf{ZT}^f(tkid).$
$\mathbf{Ticket}(tkid,'\text{ Zendesk v1 API}', f_{ZST,3}(tkid), f_{ZST,4}(tkid), f_{ZST,5}(tkid), f_{ZST,6}(tkid),$
$f_{ZST,7}(tkid), \text{Closed}') \leftarrow \mathbf{ZST}^f(tkid).$
$\mathbf{ZTID}(tkid) \leftarrow \mathbf{ZST}^f(tkid).$
$\mathbf{Ticket}(tkid,'\text{ Zendesk v1 API}', tkn, tkcd, tkdd, tkcmpd, tkp, tks) \leftarrow$
$\mathbf{ZTID}(tkid), \mathbf{ZTD}^{bfffffff}(tkid, tkn, tkcd, tkdd, tkcmpd, tkp, tks).$
$\mathbf{Page}(pgno,'\text{ Uservoice v1 API}','\text{ Uservoicev1Ticket}', pgsize) \leftarrow \mathbf{UTT}^{ff}(pgno, pgsize).$
$\mathbf{UPgNo}(pgno) \leftarrow \mathbf{UTT}^{ff}(pgno, pgsize).$
$\mathbf{UPgSize}(pgsize) \leftarrow \mathbf{UTT}^{ff}(pgno, pgsize).$
$\mathbf{Page}(pgn,'\text{ Uservoice v1 API}','\text{ Uservoicev1Ticket}', pgs) \leftarrow$
$\mathbf{UPgNo}(pgno), \mathbf{UPgSize}(pgsize), \mathbf{UT}^{bbffffff}(pgn, pgs, id, tkn, tkcd, tks, tkp)$
$\mathbf{Ticket}(id,'\text{ Uservoice v1 API}', tkn, tkcd, f_{UT,5}(pgn, pgs, id, tkn, tkcd, tks, tkp),$
$tkcmpd, tkp, tks) \leftarrow$
$\mathbf{UPgNo}(pgno), \mathbf{UPgSize}(pgsize), \mathbf{UT}^{bbffffff}(pgn, pgs, id, tkn, tkcd, tks, tkp).$

This rewriting is a bit long, but we emphasize the fact that it is a real case which is described here. Now, from the previous records DNT, we can define with SQL queries the following performance indicators : *Total New Tickets Registered in a month*, *Total High Priority Tickets Registered in a month* and *Percentage of High Priority Tickets Registered in a month*. For example the performance indicator *Total High Priority Tickets Registered in a month* definition will be :
SELECT count(tkid) FROM DNT WHERE tkcdate < sysdate and tkcate > sysdate - interval '30' day AND tkpriority='High' ;

In DaWeS, user queries are performance indicators definitions. We have chosen to distinguish two kinds of performance indicators : basic ones, called records, and complex ones, called performance indicators. Indeed, even if the inverse-rules algorithm enables the user to pose recursive datalog queries, the rewriting process is not able to deal with any aggregation function, which are mandatory to define interesting performance indicators. So the idea is first to use mediation with the rewriting process to get data from WS, then to materialize these data in the database of DaWeS, and at last to query these data to generate performance indicators. Records are user queries posed over the virtual global schema, that are rewritten during mediation to query WS and to fetch their data. Performance indicators are user queries posed over the materialized schema built with the record relations. Record queries are datalog queries. Performance queries are full SQL queries, extensible to all possible OLAP operators. Since records are materialized and business performance are computed from them, these can be updated easily when new data for the underlying records are fetched. This two layers query architecture is really interesting since it allows a user to easily change a service provider while still being able to compute her performance indicators with her full dataset (including the old data from the previous providers).

4. DaWeS architecture

The basic underlying architecture of DaWeS is shown in Figure 1. The ETL part of DaWeS is made up with three components : the query rewriter, the answer builder and the generic HTTP WS wrapper. The storage of DaWeS is organized in four schemas : the global schema (virtual), the API schema (virtual), the record schema (materialized) and the performance indicator schema (materialized). The last part is the query evaluator which is given by the underlying DBMS.

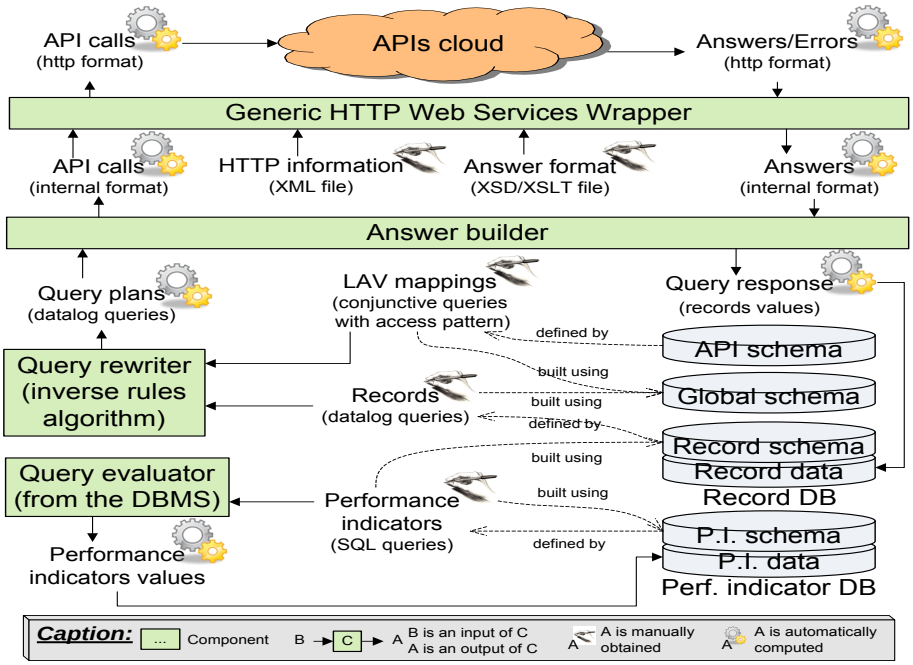


Figure 1. *DaWeS : Basic Architecture*

When a DaWeS administrator wants to add a new WS API operation, he manually gathers all HTTP information (url, authentication, ...) and all operation response formats details (to make the link with DaWeS data formats). Then he manually defines the LAV mapping and the record the operation will provide, both as queries over the global schema. The name and query of each LAV mapping (resp. record) are stored in the API schema (resp. record schema). After this, a DaWeS user can define her own performance indicator by a query over the record and/or the performance indicator schemas. The name and query of the indicator are stored in the performance indicator schema. Here, we emphasize the fact that no human intervention is devoted to programming in a high level (procedural) language (e.g. Java). Instead, only declarative languages are used here.

The automated process can then be executed. First, the query plan is computed by the inverse rule algorithm in the query rewriter from the record definition and the LAV mappings. Then the answer builder, which consists of a datalog query engine,

executes the query plan. It sends to the generic wrapper the API calls in an internal format, which consists of the atoms of the query plan, along with the authentication and input parameters. The generic HTTP WS wrapper is then used to make the WS API operation calls and transform the response in a manner understood by the answer builder. The answer builder combine these answers to get the record values which are stored in the record database. The performance indicator can at last be computed by the underlying DBMS.

The generic HTTP WS wrapper is the component dedicated to effectively execute the query plan. It is generic in the sense that it can call any API operation, provided that some parameter values are given. These values are the HTTP information (URL, method, header and body contents, authentication mechanism) and the answer format (the XSD schema of the response and XSLT translation to transform the API data formats to DaWeS desired data formats). In case of JSON message format, we translate it first to XML and then get its XSD and XSLT. As seen before, all these informations must be manually gathered when the service operations are modeled with respect to the global schema. But after this, the wrapper is able to automatically perform the right API operation call, and get back the results to DaWeS. Internally the wrapper consists of a response validator (using XSD) and a response tranformator (using XSLT). We also used cache in the wrapper in order to reduce the number of calls (the recently made operation response is cached for future use). When the answer builder requests for making an API call to the generic wrapper, the wrapper checks the cache whether the response is available and if available returns this cached response. Else the wrapper frames the HTTP header, method, body and URL for framing the API call with the input parameter values given by the answer builder. Once the response is obtained, the response validator ensures that the response schema is the same as that was registered before (else, it's a sign of possible API/operation level change). If the response is valid, the transformator makes use of the XSLT to transform the response to the desired format. This response is cached and then returned to the answer builder.

A special feature of DaWeS is that the warehouse schema, made up with the record and performance indicator schemas, is dynamic : the set of its relations will evolve in course of time, since we can add or delete new records or indicators. To handle this, we have followed a simple approach which consists of making use of only two big tables to store both the warehouse schema and its associated record and performance indicator data. This can be viewed as an extra logical layer between the schema exposed to the user and the implementation of the DBMS. It implies some small computation overhead, but ensures our warehouse schema can evolve transparently for the user.

DaWeS was tested with Intel(R) Pentium(R) Dual CPU @ 2.16GHz processor, a system memory of 3GiB and Ubuntu 13.04 (32 bits) operating system. We used Oracle 11g (11.2.0.1.0) as the database. DaWeS was developed and run using Java 1.7.0_25. We chose IRIS (Integrated Rule Inference System) (IRIS, 2008) as the datalog engine to perform query evaluation and configured it to make use of the generic HTTP WS wrapper during query evaluation in the answer builder. We chose IRIS considering its capability to handle adornments (to specify access patterns in the relations), func-

Tableau 3. *Characteristics of the Qualitative Tests*

Characteristics	Description
1. Nb of domain of WS	3
2. Nb of Web services considered	12
3. Nb of API operations considered	35 (Operations, so 35 LAV Mappings)
4. Nb of Global Schema Relations	12
5. Nb of Test Organizations	100 (homogeneous test organizations)
6. Message Formats	XML, JSON
7. Authentication Mechanisms	HTTP basic authentication, OAuth 1.0
8. Operation details	No, one or more input parameters ; pagination
9. Nb of Record Definitions	17
10. Nb of Perf. Indicator Queries	20
11. Data types	Strings, Integers, Dates, Enumerated data types

Tableau 4. *Record Definitions*

Project Management	Support (Helpdesk)	Email Marketing
Daily New Projects(1), Daily Active Projects(2), Daily OnHold Projects(3), Daily OnHold or Archived Projects(4), Daily New Tasks(5), Daily Open Tasks(6), Daily Closed Tasks(7), Daily TodoLists(8), Daily Same Status Projects(9)	Daily New Forums(10), Daily All Forums(11), Daily New Topics(12), Daily New Tickets(13), Daily Open Tickets(14), Daily Closed Tickets(15)	Daily New Campaigns(16) and Daily Campaign Statistics(17)

tional symbols (generated with the inverse rules query rewriting), built-in predicates (like equality predicate EQUAL, useful to handle functional dependencies) and the capability to refer external sources during query evaluation.

5. Experiments

We performed various qualitative and quantitative tests on DaWeS. The first step was to create realistic business-like data : we used the web interfaces of each tested service as if we were a company using it. Then we run qualitative test followed by quantitative ones. Qualitative tests aim at checking if the process computes what it is expected to. The characteristics of the tests undertaken by us is summarized in the table 5. We considered the record definitions in Table 5 created similarly as in example 2. The performance indicators considered by us are given in the Table 5.

Grounding our tests on data we had given to the WS via their associated web sites enabled us to easily check if the records and performance indicators computations were right, which was the case. Moreover, the generic wrapper was able to make API calls to any web service. Of course, these results heavily depend on the precise modeling of LAV mappings, HTTP information and answer formats. For example, if the domains of attributes, in predicates with access patterns, are not distinguished accor-

Tableau 5. *Performance Indicator Queries*

Project Management	Support (Helpdesk)	Email Marketing
Total Monthly New Projects, Total Monthly Active Projects, Total Monthly OnHold Projects, Total Monthly Completed Tasks, Average Tasks Completed Daily in a month, Total Monthly New Tasks, Total Todo Lists, Percentage of tasks completed to tasks created in a day	Daily Average Resolution Time, Total New Tickets Registered in a month, Total New Forums Registered in a month, All Forums in a month, Total New Topics Registered in a month, Total High Priority Tickets Registered in a month, Percentage of High Priority Tickets Registered in a month	Total Monthly New Campaigns, Monthly Click Throughs of Campaign, Monthly Forwards of Campaign, Monthly Bounces of Campaign, Total Monthly Solved Tickets

ding to the WS they refer to, then the query plan may imply uninteresting answers. This is the case when output data from operation 1 from service 1 are taken as input for operation 2 from service 2, just because they have the same domain of values. In example 2, if we used only the domains (PgNo, PgSize) instead of (DPgNo, DPgSize, UPgNo, UPgSize), the output page numbers and page sizes from UTT may become input to D2C resulting in unexpected answers. Secondly, the output of UTT and D2C is actually the total number of tickets and cases respectively, these have been transformed to (pgno, pgsize) combination using XSLT. Therefore it requires certain human effort to deal with such cases, where direct composition may not work. Thirdly XSD aren't usually provided in the service documentation and only example XML (JSON) responses are available for reference. It takes additional effort to create and validate XSD from the examples since often there may be discrepancies (extra or missing XML elements or attributes) in the results obtained by actual API calls and the given XML examples. Overall, the time taken to manually read the HTML documentation to declaratively describe the API is 3-4 hours.

Concerning the quantitative tests, figure 2 shows the time taken to compute the records given in table 5. Since our performance results highly depend on network communication times, the computation times were each measured 100 times so as to obtain an average time where the influence of network traffic peaks is limited. In figure 2, the total (average) time of fetching the 17 records was 104.82 seconds. Each of the 17 times are also average times. We can see that the cache implemented in the generic wrapper has a real impact on performances. Indeed, some records (2,3,4,6,7,8,9,11,14,15,17) can use the cached data from other ones (1,5,10,13,16). For example, record 14 can used the cached data of record 13 because the ticket details have already been fetched from the various services during the query evaluation of 13, transformed and cached, the transformed WS responses can as well be used for record 14. That's why we observe on the first chart in figure 2 high values for records 1, 5, 10, 12, 13, 16, and low values for the others. This point is checked also in the second chart which shows the number of API operation calls made during the query evaluation for every record definition. It shows how cache performs optimization by

avoiding the repetition of calls.

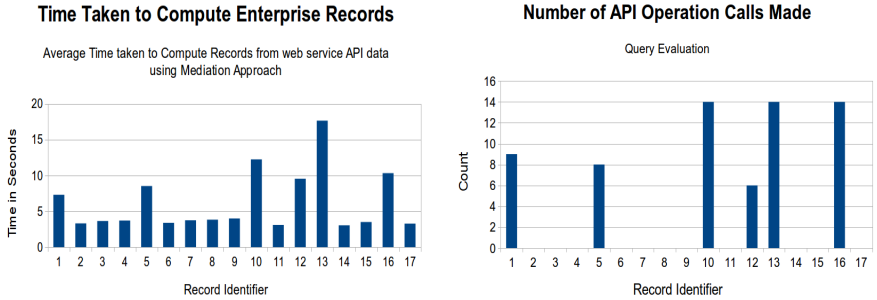


Figure 2. *DaWeS : average times to fetch record data and number of API calls.*

6. Related Works

In DaWeS, 3 important issues are handled : (i) mediation with WS as sources, (ii) generic wrapping to WS, and, (iii) mediation as an ETL tool to feed a data warehouse.

Considering (i), an important approach for performing data integration using WS is ActiveXML (Abiteboul *et al.*, 2002 ; Salem *et al.*, 2013), a language that extends XML to allow the embedding of WS calls. We think that ActiveXML could be a possible extension to DaWeS. The generic wrapper could be extended to create ActiveXML documents so that we can connect to ActiveXML services. In such an extension, we would just use the intentional part of ActiveXML documents, since using the extensional part means storing data besides calls, what we do not need (nor want) to do. So using ActiveXML in this way would be somehow contradictory with one of its main objectives which is to develop a dynamic and powerful data oriented scheme for distributed computation (e.g. peer-to-peer data integration). Indeed, DaWeS is a centralized system, since it is in fine a data warehouse. Other approaches related to (i) and closer to DaWeS than ActiveXML are surveyed in table 6.

About (ii), (Benatallah *et al.*, 2005 ; van den Heuvel *et al.*, 2007) have discussed configurable adapters (wrappers) before to deal with WS replacement and evolution. Compared to DaWeS, their main drawback is they are dependent on (so restricted to) the use of various business standards like BPEL. DaWeS is aimed at working with what is actually used to expose WS API. So focusing on one particular standard, at the exclusion of the others, is not the aim of DaWeS. DaWeS requires manual intervention for translating human readable (HTML web pages) interface description to a desired internal format (LAV Mapping, XSD and XSLT for every WS API operation). Several machine readable interface descriptions to syntactically and semantic describe the WS API operations like WSDL (W3C, 2001), WADL (Hadley, 2006) and hRESTS (Kopecký *et al.*, 2008) are useful for automatic code generation of the wrapper. An industry wide acceptance of these standards is a major concern. A generic wrapper in our context of integrating with numerous WS is easier to manage than having a wrapper for every web service. Also the generic wrapper takes as input a declarative approach to WS, making it furthermore easier to manage.

Tableau 6. *Data Integration and Web Services : State of the Art*

Characteristics	a. DaWeS	b. (Zhu *et al.*, 2004)	c. (Benslimane *et al.*, 2008)	d. (Thakkar *et al.*, 2003)	e. (Barhamgi *et al.*, 2008)
1. Primary aim	Building Data warehouse fed with WS	Large scale data integration from autonomous organizations	Mashups or Composition of two or more WS to generate new service	Mediator As a Web Service Generator	Automatic composition of data providing WS
2. Primary Targeted audience	Business enterprises using WS	Health services	Internet users	Service providers and internet users	Bioinformatics and healthcare systems
3. Underlying mechanism	Mediation approach (query rewriting)	Federated Database System	Web service composition using automated or graphical composition tools	Mediation approach (query rewriting)	Mediation approach (query rewriting)
4. Use of standards	HTTP, XML, JSON, XSD and XSLT	WSDL, UDDI, XML, DAML-S	XML, JSON, HTTP	XML, SOAP	RDF, SPARQL
5. API Operations Handled	Resource access	Resource access	Resource access and manipulation	Resource access	Resource Access
6. Algorithms used	Inverse Rules algorithm	Federated Query Services (query decomposer and query integrator)	Usually manual intervention to create the composition of services	Modified Inverse Rules algorithm	RDF query rewriting algorithm
7. User Schema	Dynamic warehouse schema	Schema generated on the fly	No schema (not needed)	Global schema	Mediated Ontology

Concerning (iii), we recall from (Trujillo et Luján-Mora, 2003) the main tasks characterizing the conceptual UML model of the ETL process : selection of the sources, transformation of the data from the sources, joining the sources to load the data for a target, finding the target, mapping the data source attributes to the target attributes and loading the data in the target. Clearly, DaWeS closely follows these requirements : the query rewriting algorithm ensures the selection and joining of the sources, the wrapper joins the sources to load the data and uses the XSLT files to perform data transformation in accordance to the target (record) schema, and the query response constitutes the data for the target. As a special feature, DaWeS enables the handling of a dynamic schema, which is transparent for the user (cf section 4). On one side, this is very convenient for the user to be able to quickly define new performance indicator. On the other side, it is less straightforward to applying popular data warehouses storage approaches, like the Inmon approach (Inmon, 1992), based on the use of normalized 3NF tables and the Kimball approach (Kimball, 1996), defining the star schema storage organization. For example, it is not clear yet what is the impact of our dynamic schema on the handling of advanced performance indicators like the CUBE operators (used along with the star schema).

7. Conclusion

The growing use of WS among the enterprises cannot be undermined. Our prototype shows it is possible to build a data warehouse fed with web services which is

aimed towards scalability and adaptability and which can be managed using declarative languages only. Mediation as an ETL approach used along with the generic HTTP WS wrapper demonstrates that the extraction of data from WS is not as complex compared to the various web scraping techniques and wrappers for textual sources and legacy databases, even with basic WS standards.

DaWeS has several other components like calibration and error handling that have not been described here due to space limitations. Records and performance indicators are periodically calibrated using various test data and WS data so as to ensure their accuracy. Calibration is used in conjunction with the error handling (errors like unexpected response format) to detect any (unannounced API change) and to trigger a manual intervention (update LAV Mapping, XSD, XSLT...). We are currently working towards a static optimization on the domain rules to reduce the API operation calls for operations that have functional dependencies on their input attributes . We want to extend DaWeS with additional set of constraints in the form of more tuple generating dependencies (TGD) than just functional dependencies. It will be further extended to a cloud infrastructure reaping the benefits of just two tables for the enterprise data.

Remerciements

We thank the Conseil General of the Region of Auvergne (France) and FEDER for funding our research project. We would also like to thank Franck Martin and Lionel Peyron of Rootsystem for their feedback during the development of DaWeS.

8. Bibliographie

Abiteboul S., Benjelloun O., Manolescu I., Milo T., Weber R., « Active XML Peer-to-Peer Data and Web Services Integration », *VLDB*, 2002, p. 1087-1090.

Abiteboul S., Duschka O. M., « Complexity of Answering Queries Using Materialized Views », *PODS*, 1998, p. 254-263.

Barhamgi M., Benslimane D., Ouksel A. M., « Composing and optimizing data providing web services », *WWW*, ACM, 2008, p. 1141-1142.

Benatallah B., Casati F., Grigori D., Nezhad H. R. M., Toumani F., « Developing Adapters for Web Services Integration », *CAiSE*, vol. 3520, 2005, p. 415-429.

Benslimane D., Dustdar S., Sheth A. P., « Services Mashups : The New Generation of Web Applications », *IEEE Internet Computing*, vol. 12, n° 5, 2008.

Burstein M. H., Hobbs J. R., Lassila O., Martin D., McDermott D. V., McIlraith S. A., Narayanan S., Paolucci M., Payne T. R., Sycara K. P., « DAML-S : Web Service Description for the Semantic Web », *Proceedings of ISWC*, 2002, p. 348–363.

Chawathe S., Garcia-Molina H., Hammer J., Ireland K., Papakonstantinou Y., Ullman J., Widom J., « The TSIMMIS Project Integration of Heterogeneous Information Sources », *Proceedings of IPSJ Conference*, 1994, p. 7–18.

Duschka O. M., Genesereth M. R., Levy A. Y., « Recursive Query Plans for Data Integration »,

J. Log. Program., vol. 43, n° 1, 2000, p. 49-73.

Duschka O. M., Genesereth M. R., « Answering Recursive Queries Using Views », *PODS*, 1997.

Fielding R. T., « Architectural Styles and the Design of Network-based Software Architectures », 2000.

Friedman M., Levy A. Y., Millstein T. D., « Navigational Plans For Data Integration », *AAAI/IAAI*, AAAI Press / The MIT Press, 1999.

Hadley M. J., « Web Application Description Language (WADL) », rapport, 2006, Sun Microsystems, Inc., Mountain View, CA, USA.

Inmon W. H., *Building the Data Warehouse*, John Wiley & Sons., New York, NY, USA, 1992.

IRIS, « Integrated Rule Inference System - API and User Guide », 2008.

Kimball R., *The Data Warehouse Toolkit Practical Techniques for Building Dimensional Data Warehouses*, John Wiley, 1996.

Kopecký J., Vitvar T., Bournez C., Farrell J., « SAWSDL Semantic Annotations for WSDL and XML Schema », *IEEE Internet Computing*, vol. 11, n° 6, 2007.

Kopecký J., Gomadam K., Vitvar T., « hRESTS An HTML Microformat for Describing RESTful Web Services », *Proceedings of the IEEE/WIC/ACM*, WI-IAT '08, IEEE Computer Society, 2008, p. 619–625.

Levy A. Y., Rajaraman A., Ordille J. J., « Query-Answering Algorithms for Information Agents », *AAAI/IAAI, Vol. 1*, AAAI Press / The MIT Press, 1996, p. 40-47.

Martin D., Paolucci M., Wagner M., « Bringing Semantic Annotations to Web Services OWL-S from the SAWSDL Perspective », *ISWC/ASWC*, 2007, p. 340–352.

Pottinger R., Levy A. Y., « A Scalable Algorithm for Answering Queries Using Views », *VLDB*, 2000, p. 484-495.

ProgrammableWeb, « http//www.programmableweb.com », December 2013.

Salem R., Boussaïd O., Darmont J., « Active XML-based Web Data Integration », *Information Systems Frontiers*, vol. 15, n° 3, 2013, p. 371–398.

Thakkar S., Knoblock C. A., Ambite J. L., « A View Integration Approach to Dynamic Composition of Web Services », *ICAPS Workshop on Planning for Web Services*, 2003.

Trujillo J., Luján-Mora S., « A UML Based Approach for Modeling ETL Processes in Data Warehouses », *ER*, vol. 2813, 2003, p. 307-320, Springer.

Ullman J. D., *Principles of Database and Knowledge-Base Systems, Volume II*, Computer Science Press, 1989.

Ullman J. D., « Information integration using logical views », *Theor. Comput. Sci.*, vol. 239, n° 2, 2000, p. 189-210.

van den Heuvel W.-J., Weigand H., Hiel M., « Configurable adapters the substrate of self-adaptive web services », *Proceedings of ICEC*, ACM, 2007.

W3C, « Web Service Description Language 1.1 », 2001.

Wiederhold G., « Mediators in the Architecture of Future Information Systems », *Computer*, vol. 25, n° 3, 1992, p. 38–49, IEEE Computer Society Press.

Zhu F., Turner M., Kotsiopoulos I. A., Bennett K. H., Russell M., Budgen D., Brereton P., Keane J. A., Layzell P. J., Rigby M., Xu J., « Dynamic Data Integration Using Web Services », *ICWS*, IEEE Computer Society, 2004, p. 262-269.

Programmation par les utilisateurs finaux : Composition d'applications Web respectueuse de la vie privée

Aurélien Faravelon[1], Eric Céret[2], Christine Verdier[1]

1. Laboratoire d'informatique de Grenoble, Équipe SIGMA
220 rue de la chimie, F-38400 Saint Martin d'Hères, France
2. Laboratoire d'informatique de Grenoble, Équipe IIHM
220 rue de la chimie, F-38400 Saint Martin d'Hères, France
{prenom.nom@imag.fr}

RESUME. Les applications web comme Facebook, Gmail ou Dropbox connaissent un large succès. Chacune propose un ensemble de fonctionnalités que l'utilisateur peut parfois composer en installant dans une application source une application tierce. La composition est ainsi un moyen, pour l'utilisateur, de bénéficier de fonctionnalités étendues. Cependant, la composition se heurte à deux problèmes. L'utilisateur ne peut pas composer à sa guise les applications puisque la composition nécessite la production de code de médiation entre les applications. Les données échangées entre les applications doivent être contrôlées pour respecter la vie privée de l'utilisateur ce qui n'est pas aujourd'hui complètement possible. Dans cet article, nous cherchons à permettre aux utilisateurs finaux de composer à leur guise leurs applications web tout en configurant le respect de leur vie privée. Pour ce faire, nous proposons une approche dirigée par les modèles de configuration des applications web et de leur composition. Cette approche est implémentée sous la forme d'un environnement de spécification et d'exécution d'une composition d'applications web. Cet environnement est utilisé sur un cas d'étude tiré de la vie réelle.

ABSTRACT. Facebook, Gmail and Dropbox are only three examples of successful web applications. Each of them provides specific functionalities and application composition allows user to extend the range of functionalities they can benefit from. However, they rely on mediation code produced by expert developers. As a result, end users cannot freely compose web applications. They cannot precisely configure their privacy preferences either. These two constrains impair users experience. Our work seeks to deal with this lacks. We propose a model-driven approach which permits end users to specify the composition of their web applications and their privacy preferences. Our approach allows to generate the execution code so that end users do not have to rely on developers. Our work is implemented as a modeling and execution tool. We use our tool to realize a real-world composition

MOTS-CLES : vie privée, application web, programmation par les utilisateurs

KEYWORDS: privacy, web applications, end-user programming

1. Introduction

Le web ressemble, pour ses utilisateurs, à un vaste système d'information distribué. Réservation de voyages, facturation, la plupart des activités peuvent aujourd'hui être exécutées en composant des applications web. Souvent, ces applications peuvent être connectées les unes aux autres. Ainsi est-il possible de se connecter à un site de réservation comme *hotels.com*[1] avec les identifiants d'un réseau social comme *Facebook*[1].

Dans le domaine des entreprises, l'exécution d'un processus intégrant un ensemble d'applications existantes est bien maîtrisée. Elle permet notamment de s'adapter aux applications disponibles et d'introduire de la flexibilité dans l'exécution des processus. L'intégration des « applications web » par et pour les utilisateurs finaux reste, elle, un domaine de recherche récent. Aujourd'hui, les applications sont composées de manière *ad hoc* : les développeurs créent des interfaces entre deux applications spécifiques et les rendent disponibles aux utilisateurs. Les utilisateurs finaux ne peuvent pas composer à leur guise les applications qu'ils utilisent, leurs fonctionnalités et leurs ressources. On peut identifier deux difficultés principales qui empêchent les utilisateurs de composer leurs applications :

- La composition d'applications web nécessite la compréhension d'API et la production de code informatique, deux opérations qui ne sont souvent pas à la portée des utilisateurs finaux.

- Le partage d'informations entre des applications fournies par des vendeurs variés pose de nombreux problèmes en termes de vie privée. Une fois l'accès aux données autorisé pour une application, il est quasiment impossible de vérifier que cette dernière n'utilisera pas les données d'une façon imprévue par l'utilisateur ou ne les revendra pas, par exemple. Des protocoles comme oAuth[1] existent aujourd'hui pour limiter l'accès d'applications tierces aux données d'un utilisateur. Cependant, l'implémentation actuelle de ces protocoles n'offre qu'un contrôle limité aux utilisateurs finaux : en général, les possibilités de paramétrage des droits d'accès sont restreintes. Ces difficultés à offrir aux utilisateurs la maîtrise de leurs données personnelles est, depuis 2010, au centre d'un vif débat entre des acteurs majeurs d'Internet (comme Facebook ou Google) et les associations de consommateurs, les organismes de contrôle (comme la CNIL en France) et même l'Union Européenne[2]. Les utilisateurs ont ainsi tendance à se tourner vers les applications les plus simples à configurer et qui leur donnent le plus de contrôle sur leurs données[3].

Dès lors, la composition d'applications web par et pour les utilisateurs finaux représente un défi. Elle permet potentiellement aux utilisateurs de tirer pleinement

---

1 voir http://fr.hotels.com, https://www.facebook.com et http://oauth.net/2

2 La justice américaine a condamné Facebook pour avoir utilisé des données personnelles dans ses publicités. Début 2014, en France, Google a été doublement condamné par la CNIL puis par le Conseil d'Etat. Voir le dossier de l'Express du 07/02/2014.

3 Voir le récent reportage de *Libération* à ce sujet : http://tinyurl.com/pz9zvlz

parti des applications qu'ils utilisent et des données qu'ils partagent. Les recherches existantes n'apportent qu'une réponse partielle à ce domaine : l'ingénierie des processus métiers, surtout lorsqu'elle prend en compte des propriétés comme la protection de la vie privée, s'adresse à des experts, les concepteurs d'applications qui ont des connaissances techniques. Le *end-user programming* (Jones 1995) se donne, lui, pour but de permettre aux utilisateurs d'étendre leurs applications. Il représente une direction intéressante mais il manque un lien entre le *end-user programming* et le niveau technique d'implémentation des applications. Implémenter une composition d'applications respectueuse de la vie privée spécifiée par un utilisateur final nécessite ainsi de trouver un langage commun aux utilisateurs finaux et aux développeurs.

Cet article présente une adaptation de l'approche centrée sur la protection de la vie privée dans les architectures orientées services de (Faravelon *et al.* 2012). Cette approche propose de spécifier la vie privée au niveau conception et de générer la SOA qui permet d'exécuter une application sécurisée et respectueuse de la vie privée. Nous adaptons l'organisation en deux niveaux, spécification et configuration/implémentation pour permettre (a) aux développeurs de décrire l'API proposée pour leurs applications, (b) aux utilisateurs de décrire la composition des API des applications qu'ils ont choisies et les droits d'accès aux ressources associés à cette composition et (c) de générer la couche applicative d'exécution correspondante.

L'article est organisé de la manière suivante. La Section 2 introduit notre cas d'utilisation. Dans la Section 3, nous dressons un état de l'art afin de montrer les manques que l'on peut identifier dans les recherches actuelles. Nous présentons ensuite notre approche d'un point de vue global dans la Section 4 avant de détailler dans la Section 5 les métamodèles que nous proposons et les outils de configuration automatique des applications web. Enfin, dans la Section 6, nous présentons l'implémentation de notre approche et son application à notre cas d'utilisation avant de discuter notre travail et de tirer des perspectives de recherche dans la Section 7.

2. Cas d'utilisation

Nous prenons dans ce papier l'exemple d'un procédé d'impression en ligne d'un ensemble de photographies, présenté sur la Figure 1. Les applications nécessaires à sa réalisation existent déjà. Les photographies d'un utilisateur peuvent être extraites d'un réseau social comme *Facebook* et retouchées grâce à un service en ligne comme *Pho.to*[6]. Enfin, des applications d'impression en ligne comme *Pwinty*[4] permettent d'imprimer les photos et de facturer l'impression. Cependant, la réalisation du procédé se heurte aux difficultés suivantes :

- Chaque application est proposée par un fournisseur différent. L'utilisateur doit ainsi s'authentifier auprès de chaque application soit à l'aide d'un compte propre à l'application, soit avec un compte associé à une autre application.

[4] voir http://pho.to/editor-platform et http://www.pwinty.com/Overview

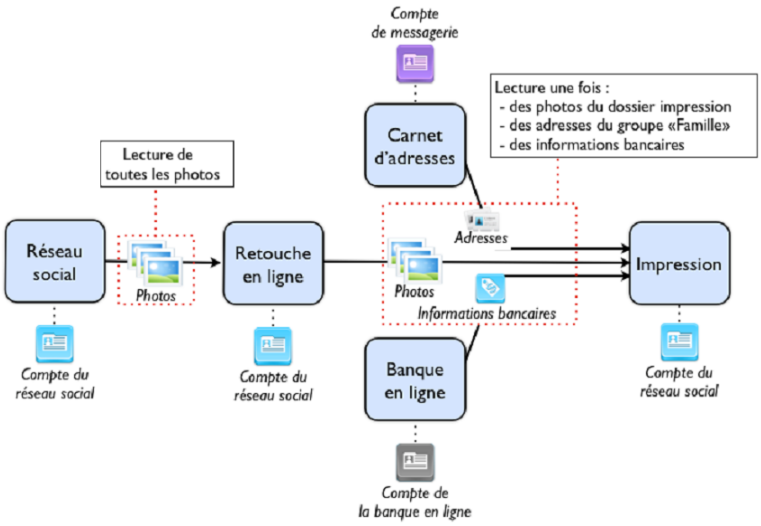


Figure 1. Un exemple de processus intégrant des applications web

- Les applications manipulent des types communs de ressources (dans cet exemple, toutes les applications traitent des images) qu'elles représentent d'une manière qui leur est propre au travers d'un vocabulaire spécifique.

- Le procédé manipule des données intimes – les photographies – et sensibles – les informations bancaires. L'utilisateur doit pouvoir contrôler précisément l'accès aux données et la durée de cet accès. Cette possibilité n'est pas offerte actuellement par les applications web,

- La liaison entre les applications est aujourd'hui réalisée de manière *ad hoc* par les développeurs. Par exemple, il n'existe pas de connecteur entre *Facebook et Pho.to*. L'utilisateur final ne peut pas lui-même connecter une application à son réseau social s'il n'existe pas de connecteur.

- Enfin, l'utilisateur est habitué à une présentation orientée ressources dans laquelle il réalise un processus par des enchaînements de tâches, sans que ce processus soit présenté explicitement sous forme de graphe d'activités.

Dans le cadre de notre travail, nous cherchons à répondre à ces quatre difficultés afin de permettre l'ingénierie de processus intégrant des applications web tout en prenant en compte la protection des ressources des utilisateurs finaux. La section suivante présente les travaux existants dans ce domaine.

3. État de l'art

Aujourd'hui, les applications web sont composées de manière *ad hoc* : les utilisateurs finaux peuvent connecter une application à une application tierce parmi celles proposées par les développeurs. Nous revenons dans cet état de l'art sur la

difficulté de composer ces applications et les enjeux de la prise en compte de la vie privée.

3.1 Composition de services web

La composition de services web repose le plus souvent sur la réalisation d'un processus spécifié à l'aide du *Business Process Modeling* Language (BPMN) ou du *Business Process Execution Language* (BEPL). Cependant, dans le cadre de l'utilisation grand public du web, les services sont réalisés sous la forme de services *Representational State Transfer* (REST) définis notamment dans (Fielding *et al.*, 2002). REST est un "style architectural" qui constitue une abstraction de la structure du World Wide Web et l'envisage comme un système distribué de gestion de ressources. Une ressource est une entité qui peut être nommée – comme une photo, une personne ou un compte bancaire (Fielding *et al.*, 2002). Si les services REST sont considérés comme des services web, l'importance de la notion de ressources dans REST le distingue des architectures orientées services (SOA) habituelles, telles qu'elles sont définies dans (Papazoglou 2003) notamment. Les SOA mettent en effet le plus souvent l'accent sur les fonctionnalités offertes par les services, leur sélection et leur invocation alors qu'une architecture REST met l'accent sur la recherche et le traitement d'un ensemble de ressources.

À cause de la différence d'architecture et d'implémentation entre REST et les SOA, l'intégration de services web et de service REST n'est pas triviale et nécessite l'extension de langages comme BEPL, comme l'ont proposé différents travaux (Wu *et al.* 2013) (Rosenberg *et al.* 2008) (Pautasso 2009). En effet, BEPL repose sur des protocoles de description des services web comme le *Web Service Description Language* (WSDL) ou de communication comme le *Simple Object Access Protocol* (SOAP) qui ne sont pas utilisés dans les architectures REST.

Enfin, ces extensions ne sont destinées qu'aux concepteurs d'applications et pas à leurs utilisateurs finaux. En effet, ces derniers sont habitués aux applications web comme un ensemble de pages qui présentent des ressources qu'ils peuvent consulter ou modifier grâce à des liens. La présentation de la composition de services REST prend ainsi souvent la forme de l'association du contenu offert par un ensemble de services présenté sous la forme d'une page web appelée *mashup*. (Hsieh *et al.* 2011) montre ainsi que la composition de services REST se fait au travers d'API et de l'utilisation de ressources identifiées par leurs URI. Les auteurs remarquent que le développement d'applications à partir de services REST est aujourd'hui faite de manière *ad hoc* : un développeur construit une application à partir d'une API existante. Contrairement aux architectures orientées services, il n'existe ainsi pas pour les architectures REST l'équivalent d'un langage abstrait de composition de services comme celui proposé par (Dami *et al.* 1998).

D'autres paradigmes d'agrégation existent, comme les agrégateurs de flux *Really Simple Syndication* (RSS). Yahoo!Pipes est un exemple de ces agrégateurs. Ces techniques ne permettent que de consulter le contenu résultant d'un ensemble de

services REST et pas de modifier ce contenu, par exemple en publiant un nouveau message ou une image.

3.2 Prise en compte de la vie privée dans les applications web

La composition de services web repose sur le partage d'informations potentiellement sensibles entre des applications fournies par des développeurs différents. Dans la mesure où il est difficile de contrôler ce que font les applications tierces des données une fois qu'elles en ont obtenu l'accès, le contrôle de l'accès aux ressources est crucial. Dans les architectures REST actuelles, ce contrôle repose sur des protocoles dédiés à la sécurité qui permettent d'authentifier les applications qui souhaitent accéder aux données d'un utilisateur et de restreindre leur accès à ces données. Oauth est le protocole le plus utilisé aujourd'hui. Le plus souvent, lorsqu'un utilisateur souhaite donner l'accès à ses données à une application tierce, une fenêtre apparaît et lui demande confirmation. La Figure 2 présente un exemple de fenêtre sur *Gmail*[5].

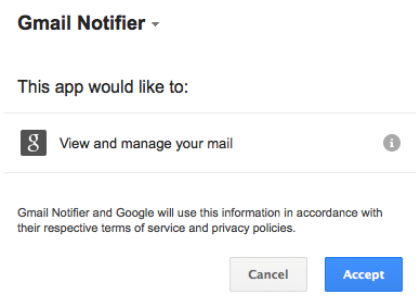


Figure 2. Un exemple de fenêtre proposée lors de l'installation d'une application tierce sur Gmail

L'efficacité de ces protocoles repose cependant sur la façon dont ils sont implémentés (Sun *et al.* 2011). En effet, certaines implémentations permettent d'usurper l'identité d'une application ou de forger des droits d'accès frauduleux. La phase de recueil du consentement est elle aussi cruciale – elle permet à l'utilisateur de savoir avec qui il partage ses données et à quelles conditions. Néanmoins, cette phase est implémentée de manière extrêmement variable. L'utilisateur ne dispose ainsi pas nécessairement de toutes les informations requises pour déterminer à quelles informations une application tierce souhaite accéder ni sa politique de conservation, de vente ou de traitement des données recueillies. Le Tableau 1 compare les informations données à l'utilisateur lors de l'installation d'applications tierces dans cinq applications web parmi les plus utilisées.

[5] voir http://pipes.yahoo.com/pipes, http://oauth.net/2 et http://gmail.com

Toutes les applications utilisent le même protocole d'autorisation – oAuth version 2.0. Néanmoins, oAuth 2.0 n'est qu'une spécification, chaque application l'implémente d'une manière propre. Toutes les applications permettent aux utilisateurs de les connecter à des applications tierces à l'installation desquelles l'utilisateur doit consentir, le plus souvent à partir d'une fenêtre dédiée. Aucune application ne permet aux utilisateurs de n'accepter qu'une partie des permissions demandées par l'application ou de restreindre les ressources auxquelles l'application souhaite accéder. La description de ces informations est importante pour permettre

Application	**Deezer**	**Facebook**	**Google**	**Twitter**	**GitHub**
Authentification	Oauth 2.0				
Documentation	http://tinyurl.com/888a82e	http://tinyurl.com/phcf7kg	http://tinyurl.com/phcf7kg	http://tinyurl.com/888a82e	http://tinyurl.com/nbp3zyr
Consentement de l'utilisateur nécessaire à l'installation	Oui				
Identité de l'application tierce	Nom, logo				
Affichage de la description de l'application tierce	Non				oui (au choix du développeur)
Contact du fournisseur	Non		email et site Web	Site web	
Actions possibles	Accepter / Refuser l'installation				
Énoncé des permissions	Enoncé des objets pouvant être lus.		Énoncé de ce que l'application peut faire et ne pas faire.		Énoncé de ce que l'application veut faire.
Niveau de détail dans la définition	Faible Utilisation de termes vagues, pas de description précise des ressources. Pas de précision de ce que l'app peut faire.		Fine Précision des objets auxquels l'application tierce veut accéder et des opérations qu'elle veut réaliser.	Moyenne Précision des objets mais pas des opérations.	Faible Pas de description des objets auxquels l'application veut accéder.
Autres informations	Non	Politique de vie privée	Non		

Tableau 1. Prise en compte de la vie privée dans cinq applications grand public

aux utilisateurs de choisir de connecter les applications. Le plus souvent, elle est vague. Par exemple, l'utilisateur est informé qu'une application veut accéder à son « profil public » sans se voir expliquer ce que ce profil contient. Enfin, alors que toutes ces applications sont « sociales » dans la mesure où elles reposent sur les interactions entre leurs utilisateurs, aucune ne propose lors de l'installation une évaluation de l'application issue de la communauté des personnes qui l'utilisent déjà, ce qui est connu pour augmenter la confiance des utilisateurs[6].

3.3 Ingénierie dirigé par les modèles, sécurité et abstraction dans les processus

Les applications web manipulent des types de ressources identiques. Par exemple, les réseaux sociaux et les systèmes de courrier électronique manipulent tous des messages électroniques. Cependant, chaque application maintient une représentation propre de ces ressources et repose sur une implémentation spécifique de l'authentification et du contrôle d'accès. La liaison entre les différentes applications se fait ainsi le plus souvent de manière *ad hoc* : les développeurs créent une application tierce pour une application source donnée et doivent réaliser le code nécessaire à chaque application source qu'ils souhaitent utiliser.

Dans les SOA, la composition abstraite de services, c'est-à-dire en se concentrant sur les fonctionnalités recherchées et non pas sur leur implémentation, peut être réalisée à partir d'une approche dirigée par les modèles (Chollet *et al.* 2008). Un métamodèle fournit le vocabulaire nécessaire à la capture d'un processus de manière abstraite. Des transformations de modèles permettent de générer le code nécessaire à la composition des services. L'ingénierie dirigée par les modèles permet aussi de définir à un niveau abstrait des propriétés de sécurité et une politique de contrôle d'accès et de générer à l'exécution le code nécessaire à la mise en œuvre de ces propriétés (Chollet *et al.* 2008) (Faravelon *et al.* 2012) (Wolter *et al.* 2007).

Néanmoins, toutes ces approches sont dirigées vers les concepteurs d'applications informatiques et non les utilisateurs finaux. (Cortes-Cornax 2011) montre ainsi que les langages d'expression de processus sont trop complexes pour être utilisés par des utilisateurs peu experts. L'auteur montre aussi que la distinction de plusieurs niveaux d'abstraction dans un métamodèle permet d'offrir un vocabulaire adapté à l'expertise de chaque utilisateur. Enfin, l'auteur montre que la syntaxe concrète d'un métamodèle est un élément crucial dans son utilisation. Dans le cas du *end-user programming* (Jones 1995), il faut ainsi proposer aux utilisateurs finaux une syntaxe graphique simple qui illustre un nombre de concepts restreint.

3.4 Conclusions à l'état de l'art et difficultés à résoudre

Notre état de l'art permet de dresser les conclusions suivantes : (1) la protection des données est indispensable pour l'acceptabilité de leur partage, mais sa mise en

[6] F-Secure. “Digital lifestyle survey”, 2013. http://www.f-secure.com

œuvre n'est pas assurée aujourd'hui de façon adéquate; (2) la composition de services REST est réalisée de manière *ad hoc* car elle repose sur des langages propres. Elle est ainsi inaccessible aux utilisateurs finaux ; (3) la composition de services REST repose sur le partage de ressources potentiellement sensibles. Néanmoins, les possibilités de contrôle d'accès des utilisateurs finaux à leurs ressources sont limitées.

Ces difficultés empêchent les utilisateurs finaux de considérer le web comme un véritable « système d'information distribué » (Fielding *et al.*, 2002) et de manipuler leurs ressources à leur guise. L'ingénierie dirigée par les modèles est un moyen efficace de composition de services et de configuration des préférences de partage mais elle n'a pas été appliquée au cas des services REST. C'est ce que nous cherchons désormais à faire.

4. Approche globale

Notre objectif est de permettre aux utilisateurs finaux des applications web de manipuler à leur guise leurs ressources. Ceci signifie permettre aux applications de partager les ressources qu'elles manipulent. Dans la mesure où ces ressources sont potentiellement sensibles, il est nécessaire de permettre aux utilisateurs finaux d'en contrôler finement l'accès.

Pour y parvenir, et afin de répondre aux difficultés identifiées dans l'état de l'art, nous proposons une approche dirigée par les modèles qui vise à permettre aux utilisateurs finaux de partager leurs ressources entre les différentes applications qu'ils utilisent sans être contraints à réaliser du code. Pour ce faire, il est nécessaire de disposer d'une description des applications à composer et de leurs besoins d'accès à un ensemble de ressources. Cette description est rédigée par les développeurs des applications. Il faut ainsi réconcilier le point de vue des utilisateurs finaux et celui des développeurs.

Afin d'atteindre ces buts, nous proposons une approche générative qui permet aux utilisateurs finaux de spécifier la composition d'un ensemble d'applications et les droits d'accès qui s'appliquent au partage de leurs ressources. La structure de notre approche est présentée sur la Figure 3. Nous fournissons un métamodèle qui donne le vocabulaire nécessaire pour exprimer de manière abstraite les ressources que l'utilisateur souhaite manipuler, les actions qu'il souhaite réaliser sur ces dernières et les droits d'accès aux ressources. Ce métamodèle est séparé en deux vues, une vue ressources et une vue autorisation.

Dans la mesure où ce métamodèle doit pouvoir être instancié par des utilisateurs finaux et des fournisseurs d'applications, nous avons défini deux niveaux d'abstraction. Le niveau d'abstraction le plus élevé est destiné aux utilisateurs finaux. Il laisse de côté les détails techniques. Le niveau d'abstraction le moins élevé est, lui, destiné aux fournisseurs d'applications. Il leur permet de décrire de manière détaillée leurs applications.

Enfin, nous avons souligné que les utilisateurs finaux étaient habitués à manipuler leurs applications au travers d'environnements graphiques orientés ressources. Afin de favoriser l'acceptabilité de notre proposition, nous définissons une syntaxe concrète graphique de notre métamodèle. Cette syntaxe concrète est inspirée des applications web afin d'être cohérente avec l'expérience habituelle des utilisateurs finaux.

Le code nécessaire à la communication entre les différentes applications et au contrôle de leur accès aux ressources est généré à partir des modèles établis par les utilisateurs finaux. L'utilisateur final n'a ainsi pas à rédiger de code informatique. Nous présentons dans les sections suivantes notre métamodèle ainsi que l'implémentation à laquelle notre travail a donné lieu.

5. Approche détaillée

Nous présentons dans cette section les détails de notre niveau spécification et de notre niveau configuration/implémentation.

5.1 Niveau d'abstraction du métamodèle pour les utilisateurs finaux

Les utilisateurs finaux n'ont pas de compétences techniques. Nous proposons un métamodèle construit à partir du vocabulaire qu'ils manipulent habituellement.

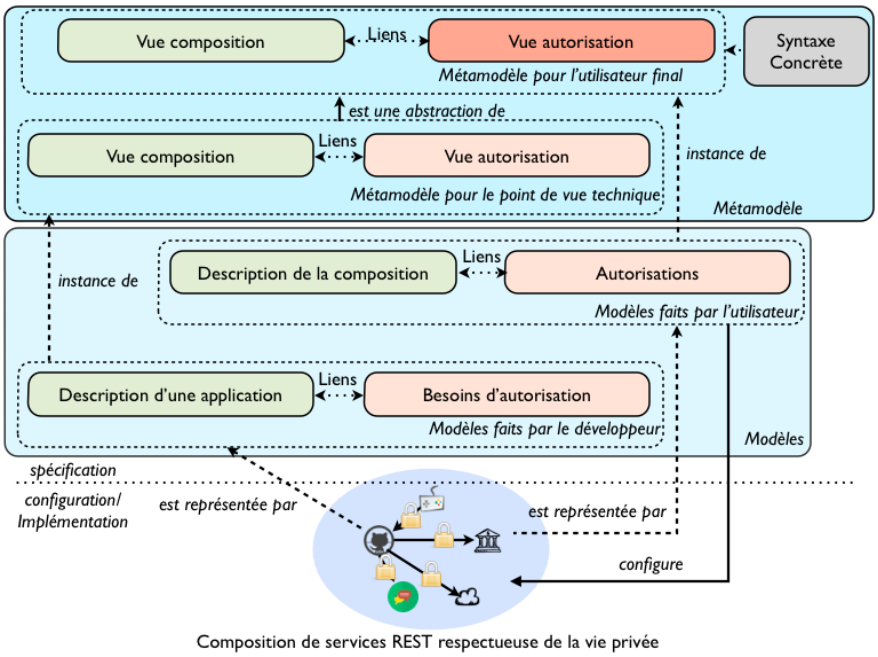


Figure 3. Approche globale

présenté sur la Figure 4.

5.1.1 Entités du métamodèle

Le métamodèle destiné aux utilisateurs, présenté sur la Figure 4, repose sur les entités suivantes : une **ressource** est un objet que l'utilisateur peut créer ou modifier. Une photographie ou un message électronique, par exemple, sont des ressources. Chaque ressource a un type. Une **application** est un programme informatique pré-existant qui permet à un utilisateur de créer ou modifier des ressources. Un site de réseau social ou un système de messagerie sont des exemples d'applications. Pour une ressource donnée, l'application qui permet de la créer est une application source. Les applications qui souhaitent accéder à cette ressource sont des applications tierces. Une **action** est un modification apportée à une ressource par une application. Des actions peuvent dépendre les unes des autres. Par exemple, l'impression d'un album photo nécessite de payer l'impression. Une application possède une politique de vie privée qui spécifie les données auxquelles elle accède et la manière dont elle les traite. Concrètement, cet attribut contient l'URL de la politique de gestion de la vie privée de l'application, ce qui permet de la télécharger dynamiquement. Une application est offerte par un **fournisseur** identifié par son nom et qui peut être contacté. Un utilisateur se connecte à une application grâce à un **compte**. Un compte est défini par un identifiant et un mot de passe. Une application peut gérer l'identification des utilisateurs pour d'autres applications.

Enfin, il est nécessaire de contrôler l'accès d'une application tierce aux ressources de l'utilisateur. Le contrôle s'effectue par la définition de **permissions** qui possèdent une durée de vie déterminée. Une permission est aussi définie par un ensemble de contraintes sur les **propriétés** des ressources considérées. Par exemple, il est possible de préciser qu'une application ne peut accéder qu'aux photos d'un album donné.

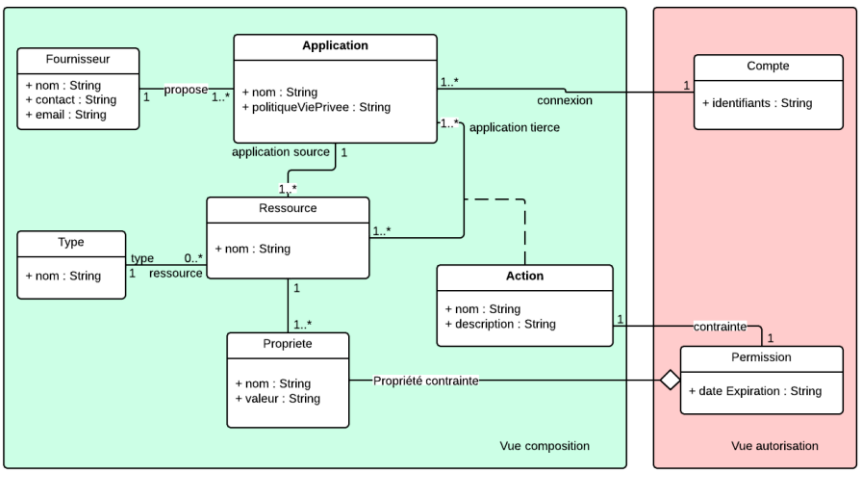


Figure 4. Métamodèle destiné aux utilisateurs finaux

5.1.2 Syntaxe concrète du métamodèle

Les utilisateurs finaux ont l'habitude d'interagir avec les applications web au travers d'environnements graphiques. Pour cette raison, il est nécessaire de définir une syntaxe concrète à notre métamodèle. Afin de favoriser l'acceptabilité de notre proposition, nous définissons cette syntaxe concrète à partir d'éléments graphiques déjà existants. Le Tableau 2 présente la syntaxe concrète que nous proposons.

5.2 Métamodèle pour les développeurs

Afin de décrire leurs applications, les développeurs ont besoin d'un vocabulaire technique. Nous reprenons ce vocabulaire dans un métamodèle présenté sur la Figure 5.

Ce métamodèle est construit en étendant le métamodèle des utilisateurs finaux avec les concepts destinés aux développeurs, qui sont présentés en gras sur la figure 5. Ces concepts sont issus des notions de l'architecture REST que l'on trouve notamment dans (Fielding *et al.* 2002). Chaque ressource possède une adresse ou URI, qui joue le rôle d'identifiant. Chaque ressource possède des **représentations**, des instances de la ressource. Par exemple, le concept d'image peut avoir plusieurs instances pour des images réelles différentes. Une représentation est décrite par un ensemble de **métadonnées**. Les types des ressources sont associés à des types MIME, ce qui permet d'identifier les équivalences entre les différents types (ex. : image et photo sont tous les deux des types MIME image). Les ressources sont regroupées dans une **API**, qui implémente une Application. Les applications pouvant avoir des niveaux de détails différents pour décrire les ressources, des modèles pivots et des adaptateurs peuvent être implémentés pour unifier ces

Entité du métamodèle	Syntaxe concrète	Exemple
Application	Logo de l'application	
Fournisseur	Texte ou icône	Crédit Agricole
Compte	Icône	
Ressource	Texte ou icône	
TypeRessource	Texte ou icône	Photos
Action	Texte ou icône	Impression
Permission	Texte	Lecture une fois

Tableau 2 - Syntaxe concrète du métamodèle pour les utilisateurs finaux

descriptions. Une action est considérée comme une action HTTP, et donc spécialisée en **Get**, **Post** ou **Delete**. Le compte d'un utilisateur peut être associé à une **autorité centrale** dans le cas où il peut être utilisé par plusieurs applications. Chaque application peut être associée à une ou plusieurs **méthodes d'authentification**. Par exemple, une application peut à la fois permettre de s'authentifier en utilisant oAuth et un login et un mot de passe.

6. Implémentation

Afin de démontrer la faisabilité de notre approche, nous avons développé un prototype de gestion de ressources distribuées entre les applications d'un utilisateur. Cet environnement permet aux utilisateurs de configurer le partage de leurs ressources et les droits d'accès à ces dernières grâce à une page web. La Figure 7 présente une capture d'écran de notre outil.

Le menu de gauche ❶ reprend la liste des ressources qu'un utilisateur peut manipuler. Cette liste est générée à partir du métamodèle pour les utilisateurs finaux. À chaque ressource est associé un menu contextuel qui permet de sélectionner sa source – par exemple un réseau social. Les ressources sélectionnées par l'utilisateur ❷ ❸ apparaissent dans l'espace central de la page web. L'utilisateur peut alors filtrer un sous-ensemble des ressources, par exemple un album auquel des photos doivent appartenir. Les ressources sélectionnées sont associées à un menu contextuel qui

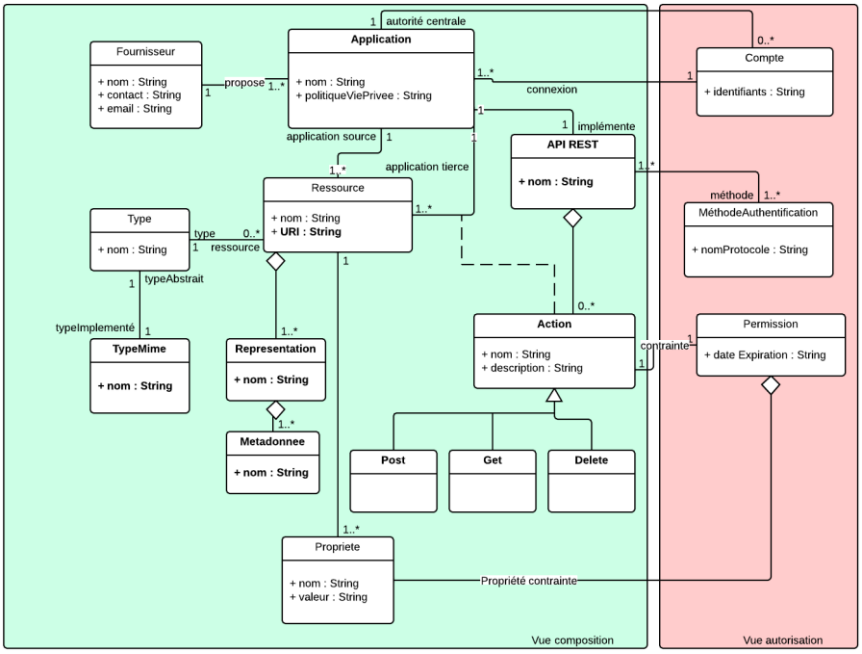


Figure 5. Métamodèle pour les développeurs

Figure 7. Capture d'écran d'une fenêtre de configuration des droits d'accès

présente les actions possibles sur un ensemble de ressources. L'enchaînement des actions permet implicitement de réaliser un processus, comme celui de notre cas d'étude.

Lorsqu'un utilisateur sélectionne une action, une boîte de dialogue ❹ de demande de confirmation de l'utilisation d'une application tierce apparaît. La boîte de dialogue présente toutes les informations spécifiant les autorisations et leurs limites ❺, ainsi que le contact chez le fournisseur de l'application tierce ❻ et les informations sur les contacts de l'utilisateur qui utilisent déjà l'application ❼. Cette boîte de dialogue est générée à partir de la description des applications fournies par les développeurs.

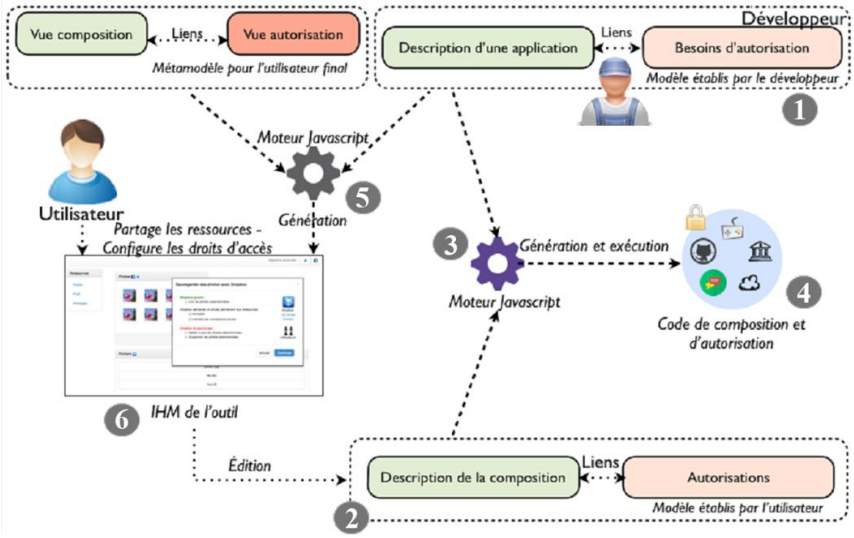


Figure 6. Principe de notre outil

La spécification des actions à effectuer sur un ensemble de ressources et des droits d'accès permet de générer le code de composition des applications et le code d'autorisation. Afin de mettre en œuvre les droits d'accès, notre prototype joue le rôle de médiateur entre les applications sources et les applications tierces : pour chaque ressource, il génère une URL vers laquelle l'application tierce pointera. Cette URL décrit les droits d'accès de l'application tierce. Avant de donner l'accès à l'application source, le prototype évalue les droits d'accès. La Figure 6 reprend le principe de notre prototype et de l'utilisation de la génération automatique. A partir des modèles des applications et des besoins d'autorisations décrits par les développeurs, un premier moteur Javascript génère l'IHM présentée à l'utilisateur final (figure 6), sur laquelle ce dernier peut définir les autorisations pour la composition concrète qu'il est en train de spécifier. A partir de cette configuration définie par l'utilisateur final, un second moteur Javascript génère le code de composition des applications et de contrôle d'accès. C'est à ce niveau qu'intervient l'originalité de notre proposition, puisque ce code est actuellement créé par les développeurs pour une composition donnée, et non généré dynamiquement.

7. Conclusion et perspectives

Les débats actuels sur la protection des données personnelles montre l'importance de la question de la vie privée. Nous avons cherché à permettre aux utilisateurs finaux des applications web de composer leur applications tout en protégeant leur vie privée. Nous entendons ici la protection de la vie privée comme le contrôle des applications tierces qui peuvent accéder aux informations d'un utilisateur et leurs droits d'accès à ces informations.

Pour y parvenir, nous avons proposé une approche dirigée par les modèles afin de configurer les applications web qu'un utilisateur souhaite composer et leurs droits d'accès à un ensemble de ressources. Cette approche permet à un utilisateur final de décrire le partage de ses ressources entre ses applications et leurs droits d'accès. Des transformations automatiques permettent de générer le code nécessaire à l'exécution de la composition et à la protection de sa vie privée.

Notre prototype se positionne comme une recommandation, dans la mesure où, pour être efficace, il faudrait qu'il soit intégré directement dans les applications Web. Dans les travaux, il conviendrait d'évaluer l'acceptabilité de cette proposition par les fournisseurs d'applications et par les utilisateurs finaux.

Notre approche est implémentée sous la forme d'un environnement de gestion des ressources d'un utilisateur. Cet environnement démontre qu'il est possible d'améliorer la prise en compte de la vie privée dans les applications web. Son fonctionnement pourrait être intégré aux plateformes existantes qui proposent déjà de partager leurs ressources avec des applications tierces. Nous prévoyons, à titre de travaux futurs, d'étendre notre approche en prenant en compte des ressources complexes, comme une structure composée d'une photo, de son auteur et de sa localisation. Nous augmenterons aussi la gestion de la cohérence des permissions de

données en définissant une sémantique formelle. Enfin, nous prévoyons d'évaluer l'utilisation de cet outil par des utilisateurs finaux.

Remerciements

Nous remercions Stéphane Frénot pour ses relectures et ses conseils.

Références

Budan Wu, Rongheng Lin, Junliang Chen (2013). Integrating RESTful Service into BPEL Business Process on Service Generation System. Proceedings of the *IEEE Service Computing Conference 2013, 2013, IEEE Computer Society.*

Cortes Cornax M. (2011). Service choreographies through a graphical notation based on abstraction layers and viewpoints. *IEEE RCIS 2011. 2011.*

Chollet, S., Lalanda, P. (2008). Security Specification at Process Level. *IEEE Service Computing Conference 2008, 2008, IEEE Computer Society.*

Dami, S., Estublier, J., Amiour, M. (1998). Apel: A Graphical Yet Executable Formalism for Process Modeling Autom. *Softw. Eng.*, vol. 5, p. 61-96.

Faravelon, A., Chollet, S., Verdier, C., Front, A. (2012). Configuring Private Data Management as Access Restrictions: From Design to Enforcement. *International Conference on Service Oriented Computing*, 2012.

Jones C. (1995). End user programming. *Computer*, vol. 28, n°.9, p.68-70

Meng-Yen Hsieh, Hua-Yi Lin, Kuan-Ching Li. (2011). A web-based travel system using mashup in the RESTful design. *International Journal of Computational Science and Engineering*. Vol. 6, n°3, p. 185-191.

Papazoglou, M. P. (2003). Service-Oriented Computing: Concepts, Characteristics and Directions. *Proceedings of the Fourth International Conference on Web Information Systems Engineering*, *2003,* IEEE Computer Society.

Pautasso P. (2009). RESTful Web service composition with BPEL for REST. *Data Knowl. Eng.*, vol. 68, n°9, p. 851-866.

Rosenberg F., Curbera F., Duftler M. J., Khalaf R. (2008). Composing RESTful Services and Collaborative Workflows: A Lightweight Approach. *IEEE Internet Computing*, vol. 12, n°5, p. :24-31.

Roy T. Fielding, Richard N. Taylor (2002). Principled design of the modern Web architecture. *ACM Trans. Internet Techn.*, vol. 2, n°2, p. 115-150.

Sun S., Beznosov K. (2012). The devil is in the (implementation) details: an empirical analysis of OAuth SSO systems. *ACM Conference on Computer and Communications Security*, *2012*. ACM.

Wolter, C., Schaad, A. (2007). Modeling of task-based authorization constraints in BPMN *BPM'07*, 2007, Springer-Verlag

Session 7

Qualité des langages et des modèles

Qualité des modèles : retour d'expériences

Sophie Dupuy-Chessa[1], Kathia Marçal de Oliveira[2], Samira Si-Said Cherfi[3]

1. Université de Grenoble Alpes, Laboratoire d'informatique de Grenoble
F-38000 Grenoble, France
Sophie.Dupuy@imag.fr

2. Laboratoire d'Automatique de Mécanique et d'Informatique Industrielles et Humaines, Université de Valenciennes et du Hainaut-Cambrésis, UMR CNRS 8201
F-59313 Valenciennes cedex 9
Kathia.oliveira@univ-valenciennes.fr

3. Laboratoire CEDRIC, Conservatoire National des Arts et Métiers
292 rue Saint Martin 75003, Paris
samira.cherfi@cnam.fr

RÉSUMÉ. Avec la complexification des systèmes d'information (systèmes ubiquitaires, entreprises ouvertes etc.), de nombreux nouveaux langages de modélisation sont proposés. Face à ce développement de langages spécifiques, on peut s'interroger sur la qualité des modèles qui en sont issus. Cet article traite de ce problème en tirant les leçons de nos expériences passées. Elles mettent en évidence les besoins d'outillage automatisé pour l'évaluation de la qualité de modèles, la participation conjointe des différentes parties prenantes dans le processus d'évaluation, et la nécessité d'envisager une véritable ingénierie des langages et des modèles centrée sur l'humain.

ABSTRACT. The increasing complexity of information systems (ubiquitous systems, open enterprises, etc.) calls for the introduction of always new modeling languages. However, the development of new domain-specific languages makes the question about their quality, as well as the quality of the produced models, an important issue for the modeling of information systems. This article deals with this issue by exposing the lessons learnt from the past experiences both on the quality of the languages and of the models expressed using them. These lessons highlight the necessity of automatic tools to support the quality evaluation of models, the collective participation of all stakeholders in the evaluation process and the definition of a human-centered language and model engineering approach.

MOTS-CLES : qualité, modèle, retour sur expérience, expérimentation

KEYWORDS: quality, model, lessons learned, experiment

1. Introduction

Depuis les années 1970, la conception d'un système d'information (SI) s'appuie presque systématiquement sur des modèles représentant le système à développer. La modélisation conceptuelle vise à acquérir et à formaliser la connaissance du domaine pour répondre aux fonctionnalités du système conçu : « l'objectif principal de la modélisation conceptuelle est la collecte et la définition formelle de la connaissance, à propos du domaine et dont le système a besoin pour effectuer les fonctions qui lui sont assignées. » (Olivé, 2007). Ainsi, le modèle conceptuel est la formalisation de l'expression des besoins et permet à ce titre de vérifier la conformité du SI au domaine (Wand & Weber, 2002). Il est également une formalisation de ce que sera le SI. Il n'est pas uniquement une description du domaine mais le support de toute la suite du développement et même de la maintenance et de l'évolution du SI (Lindland et al., 1994). De plus, de nos jours, les SI deviennent de plus en plus complexes : ils sont ubiquitaires ; les entreprises sont ouvertes, ... Les modèles, qui servent à les comprendre et à les représenter, prennent une importance accrue. D'une part, ils permettent de gérer la complexité de conception galopante de ces systèmes ; d'autre part, ils représentent de nouvelles caractéristiques du système tel que son contexte d'usage (Henricksen 2004).

Cependant, pour qu'un modèle soit vraiment utile pour le développement il faut assurer sa qualité. Dans cet article, nous faisons une analyse des travaux sur la qualité des modèles. Outre l'étude des travaux existants dans ce domaine, nous focalisons, notre réflexion sur nos expériences passées. La large couverture de nos travaux ainsi que leur complémentarité nous permettent de mettre en évidence des leçons pratiques pour aboutir à des modèles de qualité.

La section 2 présente un survol de la littérature. La section 3 introduit un résumé de nos contributions en termes de qualité des modèles. Elle traite de l'évaluation de la qualité des modèles en général, et plus particulièrement la qualité sémantique et pragmatique. Nous dressons en section 4 une synthèse de notre discussion et un bilan des leçons apprises. La section 5 présente nos conclusions et quelques perspectives de ce travail.

2. Travaux existants

La recherche ciblant la qualité des modèles conceptuels reste relativement jeune. A notre connaissance, la première approche structurant le concept de qualité des modèles a été proposée dans (Batini et al. 1992). L'étude détaillée de la littérature a permis de classer ces contributions selon deux catégories que nous détaillons par la suite : (i) les approches visant la compréhension et la caractérisation de la qualité, et (ii) les approches dédiées à la mesure et à l'évaluation de la qualité.

2.1. Compréhension et caractérisation de la qualité des modèles

La première catégorie regroupe des propositions de cadres généraux dont le but est de comprendre et de caractériser la qualité. (Lindland et al. 1994) propose d'évaluer la qualité des modèles selon trois aspects :

– **La qualité syntaxique** où l'objectif de qualité adressé est celui de la justesse du modèle vis-à-vis des concepts et des contraintes du langage. Pour atteindre cette qualité, les méthodes et les outils s'appuient sur le langage utilisé.

– **La qualité sémantique** mesure l'adéquation du modèle au domaine qu'il représente. Cette adéquation comprend l'état actuel du domaine et ses évolutions. La qualité sémantique se mesure à travers des critères tels que la complétude ou la justesse sémantique. Ces critères sont difficiles à mesurer puisqu'ils nécessitent la prise en compte de la connaissance du domaine souvent informelle et non structurée.

– **La qualité pragmatique** est liée à la compréhension d'un modèle par son audience. Il existe diverses études menées sur les facteurs pouvant avoir une influence directe ou indirecte sur la compréhension. Ces études soulignent souvent l'impact direct qu'ont la complexité et la taille de modèles sur leur compréhension. Mais la compréhension peut aussi être influencée par la documentation associée aux modèles ou par le choix de nommage des éléments du modèle.

Krogstie a étendu cette vision en ajoutant trois types de qualités : (1) la qualité physique caractérise l'évaluation du modèle auprès d'un participant ; (2) la qualité sémantique perçue permet d'évaluer la qualité sémantique d'un modèle non pas directement par rapport au domaine mais par rapport à la perception du modèle par les participants ; (3) la qualité sociale est atteinte si les différents acteurs convergent sur une représentation. Le cadre ainsi défini est appelé SEQUAL (Krogstie et al. 1995). Il intègre les acteurs, le domaine et le langage, conduisant à une vision très complète de la qualité. En 2006, Krogstie et al. ont étendu ce cadre pour intégrer aussi les modèles de processus, ce qui suppose une prise en compte de la dynamique (Krogstie et al. 2006).

D'autres travaux cherchent à définir les caractéristiques de la qualité des modèles. A titre d'exemple, citons Levitin et Redman (Levitin & Redman 1995) qui ont proposé d'organiser les caractéristiques de la qualité des modèles en six catégories (contenu, étendue, niveau de détail, composition, cohérence et réaction aux changements) ; (Marjomaa 2002) qui préconise quatre axes d'analyse de la qualité, respectivement en termes ontologiques, épistémologiques, de valeurs théoriques et pragmatiques et (Bajaj 2002) qui a étudié le critère de facilité de lecture et d'interprétation des modèles selon trois dimensions quantifiables (efficacité, efficience et simplicité d'apprentissage).

2.2 Evaluation de la qualité des modèles

La deuxième catégorie de travaux sur la qualité des modèles traite de la mesure de cette qualité au moyen de la définition de critères ou de facteurs de qualité et de métriques. La première approche structurée de cette catégorie est présentée dans (Batini et al. 1992). Pour la première fois, des critères de qualité, spécifiques aux modèles conceptuels, ont été proposés (complétude, justesse, minimalité, expressivité, lisibilité, auto-description, extensibilité et normalité). Ces critères n'ont cependant pas été associés à des métriques permettant de les mesurer.

Dans (Chidamber et Kemerer 1994) les auteurs ont développé et testé empiriquement six métriques liées à l'analyse et la conception. Moody et Shanks dans (Moody & Shanks 1994) ont défini un ensemble de métriques qui tiennent compte des catégories d'acteurs qui manipulent les modèles : utilisateurs métier, analystes de données, administrateurs de données et développeurs d'applications. Dans (Assenova et Johannesson 1996), d'autres critères tels que l'homogénéité, la taille ou la simplicité des modèles ainsi que des requêtes sur ces modèles ont été définis. De plus, les auteurs ont décrit un ensemble de transformations aidant à l'amélioration des modèles. Ce dernier aspect a d'ailleurs été rarement abordé dans les travaux sur la qualité.

Plusieurs articles discutent la qualité des modèles dans le contexte d'utilisation de l'IDM. Certains auteurs soulignent l'importance d'évaluer les modèles et les transformations en considérant des caractéristiques et mesures de qualité spécifiques pour les modèles (Mohagheghi, et Dehlen 2008, Dominguez-Mayo et al., 2010) et en assurant la traçabilité de ces caractéristiques (Oliveira, 2010). Mohagheghi et Dehlen (2009), par exemple, ont identifié six classes de caractéristiques de qualité de modèles : correction, complétude, cohérence, compréhensibilité, confinement et changeabilité. Dominguez-Mayo et al. (2010) ajoutent aussi l'importance d'évaluer le langage de transformation utilisé. (Becker 2012) propose d'utiliser les caractéristiques non-fonctionnelles dans l'analyse des modèles et transformations pour soutenir les décisions sur l'architecture du logiciel. Guana et Correal (2011), à leur tour, proposent d'évaluer les modèles de ligne de produits (dans une approche IDM) pour analyser la qualité du produit final et aussi aider la décision de choix architectural.

Finalement, concernant l'utilisation d'outils pour l'évaluation des modèles on peut citer (Vanderose et Habra, 2011) mais aussi (Rodriguez et al. , 2010) où les auteurs considèrent les aspects syntaxique, sémantique et pragmatique définis par Lindland (1994). Salvaneschi et Piazzalunga (2008) soutiennent que plusieurs modèles de qualité pour évaluer les modèles existent, mais que les modèles de logiciels ne sont pas conçus spécifiquement pour mesurer cette qualité, alors que cela est essentiel pour avoir une mesure de la qualité. (Mantyla 2004) suggère qu'à partir de plusieurs évaluations de qualité des modèles basées sur l'opinion subjective des évaluateurs, soient définies des heuristiques pour l'évaluation qui aident à la décision sur sa qualité.

En résumé, les travaux sur l'évaluation de modèles se concentrent surtout sur la définition de caractéristiques et métriques spécifiques et leur expérimentation. En outre, plusieurs méthodes (p.e, inspections, walkthrough, techniques basées sur la lecture (*reading-based tecniques*)) et outils pour l'automatisation de la collecte et interprétation des métriques sont utilisés. Dans notre travail, nous avons essayé de pallier les défauts de ces approches en alliant dans nos contributions les aspects théoriques de définition de caractéristiques/mesures ; et expérimentaux, pour leur validation et application en études de cas de systèmes, parfois issus de l'industrie.

3. Leçons sur la qualité des modèles

Nous exposons ici notre expérience dans le domaine de l'évaluation des modèles en considérant à la fois les modèles de données, modèles fonctionnels et modèles de processus.

3.1. Leçons sur la qualité des modèles de données

Pour l'évaluation de la qualité des modèles de données notre expérience couvre des critères de qualité et leur évaluation. Pour l'évaluation notre approche utilise comme fondement pratique le point de vue des professionnels. Nous avons fait participer des experts universitaires et des industriels au moyen d'enquêtes, interviews, etc.

La diversité des critères de qualité définis dans la littérature et le manque de consensus concernant leur définition rend difficile leur utilisation. En effet, les définitions sont souvent vagues ne facilitant pas le choix du critère à évaluer ni la méthode pour l'évaluer. De plus, la diversité des critères et leur multiplicité, en l'absence d'un support outillé ou de guides méthodologiques adéquats, créent une complexité rendant ces moyens d'évaluation inaccessibles.

Notre première enquête avait comme objectif de limiter notre étude aux critères de qualité pertinents pour des acteurs souhaitant effectuer des évaluations de la qualité des modèles. Nous avons donc procédé à une enquête qui consistait à recueillir les opinions des interviewés sur la qualité globale des modèles conceptuels, en plus de leurs commentaires sur un ensemble de critères de qualité extraits de la littérature (Mehmood et al. 2009). A partir des résultats de cette enquête, nous avons construit une approche s'appuyant sur des patrons de qualité dont le but est d'assister l'évaluation de la qualité en aidant à (1) identifier l'objectif de qualité recherché, (2) identifier les critères de qualité à mesurer pour atteindre ce but et enfin (3) à procéder à l'évaluation et à l'amélioration de la qualité selon les critères identifiés (Mehmood et al. 2011). Nous avons aussi mené un effort de validation de cette approche en demandant à des participants d'analyser la qualité des modèles en s'aidant des patrons de qualité. Cette deuxième expérimentation a permis d'identifier un certain nombre de leçons à considérer lors de l'évaluation des modèles :

– la difficulté de l'évaluation peut être due à une méconnaissance de la notation utilisée pour exprimer les modèles,
– la connaissance du domaine est importante pour la compréhension des modèles et par conséquent pour leur évaluation.
– les outils sont primordiaux pour aider à exploiter les connaissances concernant l'évaluation de la qualité.

Nous avons également mené une enquête sur l'évaluation de la qualité en tenant compte du profil des participants. Il s'agissait d'évaluer différents modèles conceptuels décrivant une même réalité par un ensemble d'acteurs tels que des concepteurs, des utilisateurs et des développeurs. L'enquête avait pour objectif de vérifier si certaines métriques, que nous avions définies dans notre approche pour l'évaluation de certains critères de qualité étaient en adéquation avec la perception par les parties prenantes de ces mêmes critères. Cette expérimentation a pris en compte quatre critères de qualité : clarté, minimalité, expressivité, simplicité. Pour ce faire, nous avons recueilli les avis de 113 parties prenantes, dont des professionnels des SI (87) et des utilisateurs finaux (26).

L'analyse des résultats a révélé que nos métriques sont pertinentes et peuvent être globalement validées. Elle a également révélé que les critères d'expressivité et de minimalité requièrent une compréhension de la sémantique des modèles. En conséquence, ils sont plus difficiles à comprendre. Les concepts sous-jacents ne facilitent pas l'obtention d'un consensus surtout auprès d'un public non expert en modélisation.

L'analyse a aussi permis de distinguer les utilisateurs finaux des professionnels de l'informatique dans leur perception de la qualité des modèles. Nous avons aussi pu distinguer un troisième groupe composé de concepteurs, de gestionnaires de projet, de spécialistes des systèmes d'information et, de manière surprenante, d'étudiants. Les résultats de l'évaluation par les métriques ainsi que le détail de l'expérimentation sont détaillés dans (Si-said et al 2002, Akoka et al., 2008).

A partir de la réalisation des ces expérimentations nous pouvons constater que :

– (C1) la définition des critères de qualité à mesurer et des métriques associées doit se faire en impliquant les utilisateurs et surtout les professionnels qui seront amenés à les utiliser,
– (C2) le choix de la notation utilisée impacte directement l'évaluation des modèles et pour supprimer ce biais il est important de proposer les mêmes modèles dans diverses notations selon les connaissance des acteurs chargées de l'évaluation,
– (C3) les connaissances, par les acteurs chargés de l'évaluation, du domaine sous-jacent aux modèles impacte aussi l'évaluation,
– (C4) il est important d'outiller les méthodes d'évaluation de la qualité,
– (C5) il est important de ne pas se limiter à faire des propositions de critères ou de métriques mais de valider aussi l'approche d'évaluation.

3.2. Leçons sur les modèles fonctionnels

Notre expérience d'évaluation des modèles fonctionnels et d'interaction couvre différents types de systèmes dont nous citons les principaux dans cette section. Ici aussi nos contributions allient résultats théoriques et évaluations expérimentales.

Une première expérience concerne l'évaluation des modèles conceptuels pour la définition d'un système expert dans le domaine de cardiologie (Rabelo et al., 1997). Les modèles conceptuels étaient élaborés avec la méthodologie KADS (Knowledge Acquisition and Design Structuring) spécifique à ce type de système. Des facteurs de qualité spécifiques ont été définis pour évaluer la fiabilité conceptuel ou sémantique des modèles (par exemple, exhaustivité, équivalence à une spécialiste, non redondance, nécessité) et la fiabilité de la représentation des modèles (par exemple, clarté, style, correction, uniformité du niveau d'abstraction). Chaque facteur de qualité était évalué par des inspections effectuées par les différentes parties prenantes (cardiologues, ingénieur de connaissances et développeurs). Ces différents intervenants étaient sollicités seulement sur les critères pertinents pour eux (par exemple, la correction de la notation ne pouvait pas être évaluée par des cardiologues). L'évaluation a été effectuée par trois cardiologues, deux ingénieurs de connaissances et un développeur. L'évaluation était réalisée tout d'abord individuellement puis lors d'une réunion commune de deux heures. Dans l'ensemble, les critères d'évaluation ont été considérés comme faciles à utiliser par les évaluateurs et ont aidé dans l'identification des erreurs. Cependant, les cardiologues ont signalé des difficultés dans la compréhension du modèle produit avec la méthodologie KADS et les ingénieurs de connaissances ont signalé des difficultés lors de la conception du modèle. Cette évaluation a conduit à des améliorations dans le processus de développement logiciel et la définition d'extensions pour KADS. Cette expérience nous a montré la multi-dimensionnalité de la qualité des modèles, autrement dit, qu'un modèle doit être évalué non seulement par différents critères de qualité mais aussi en considérant différents points de vue : la vision des développeurs, des experts du domaine métier (dans ce cas cardiologues) et des demandeurs (client) du projet. De plus, cette expérience nous a montré la subjectivité de l'évaluation de la qualité des modèles. Même quand nous avons essayé d'utiliser des métriques objectives, elles avaient besoin d'être associées à une évaluation qualitative de la part des experts du domaine.

Une deuxième expérience concerne l'évaluation des modèles fonctionnels utilisant diagramme de flux de données (DFD) des systèmes patrimoniaux (Ramos et al., 2004). Ces systèmes ont en général une documentation assez pauvre alors qu'une telle documentation est nécessaire à la maintenance de ces systèmes patrimoniaux. Aussi des métriques spécifiques ont été définies pour évaluer dans quelle mesure les modèles sont documentés, leur facilité de compréhension et la cohérence entre les différents modèles composants la documentation (modèles de données, modèles métier s'il existe, et diagrammes DFD de différents niveaux). Des évaluations expérimentales ont été faites utilisant la documentation des grands systèmes d'une institution bancaire. Les modèles de dix systèmes ont été évalués par trois évaluateurs ayant un profil d'analyste de système. Les évaluateurs n'avaient pas

de connaissances sur les systèmes et n'ont eu aucun contact avec les utilisateurs ou les mainteneurs du système, ainsi les résultats sont basés seulement sur l'évaluation des modèles, sans aucun biais. Les évaluations ont duré entre 48 min et 1h20. Les résultats indiquent que la qualité du niveau de documentation était faible, car malgré l'existence des diagrammes, il n'y avait pas de descriptions de leurs éléments (ex : attributs, flux, etc.). Cependant la majorité des diagrammes était considérée comme facile à comprendre, sauf pour l'un des systèmes qui a été considéré complexe. Finalement, des problèmes de cohérence entre les modèles (diagramme de contexte X DFD, modèle de donnée X DFD) ont été identifiés. Cette expérience nous a montré la difficulté d'établir des seuils pour les métriques objectives et aussi d'interpréter les résultats. La présence et l'évaluation subjective d'un expert du domaine métier étaient toujours demandées. De plus, l'expérience a confirmé le fait reconnu de faible qualité des documentations des systèmes patrimoniaux.

Une dernière expérience concerne l'évaluation de modèles de navigation développés avec la méthode OOHDM (Object-Oriented Hypermedia Design Method) pour la conception de sites web (Nery et al., 2006). Différentes évaluations de projets réels de sites ont été effectuées en considérant des métriques appropriées comme le *fan-in* et *fan-out* entre contextes de navigation, la densité de structure des navigations, la moyenne de chemin plus court, etc. Pour réaliser ces évaluations, un outil a été développé pour calculer automatiquement les mesures à partir des diagrammes élaborées en OOHDM. Ces expériences nous ont montré que les évaluations réalisées sur des modèles très proches de leur implémentation finale peuvent fournir un aperçu réel de la qualité du système final produit (dans ce cas, des sites web). Cependant, ce constat est peut être spécifique à des applications web où le codage peut correspondre directement aux navigations projetées.

La principale leçon de toutes ces expériences est qu'évaluer des modèles n'est pas facile à mettre en œuvre, et requiert la participation de différentes parties prenantes (C6) : experts du domaine métier, client, chef de projet, développeurs, etc. Cette participation est essentielle pas seulement pour la réalisation des évaluations en elles-mêmes (en utilisant des techniques spécifiques comme inspection et *walkthrough*) mais aussi pour l'interprétation des évaluations pour la prise de décision. Pour certains facteurs de qualité, cette interprétation peut être plus facile (ex. : la correction syntaxique des modèles). Mais, pour la grande majorité des facteurs de qualité l'interprétation et la décision sur la qualité finale peuvent être compliquées et dépendre de différents aspects, comme par exemple : le langage utilisé pour la modélisation, le type de projet, la connaissance sur le domaine, et surtout le besoin des utilisateurs qui est vraiment connue seulement en fin de projet et non dans la phase de conception où les modèles sont élaborés. Aussi la présence d'un expert est nécessaire pour l'interprétation et la décision sur la qualité (C7).

3.2.3. Leçons sur les modèles de processus

Notre expérience en matière de qualité des modèles de processus a été à la fois théorique et expérimentale.

Du point de vue théorique, dans (Ceret et al 2013), nous sommes allés plus loin que les travaux existants pour la comparaison des modèles de processus en proposant une taxonomie de ces modèles : nous avons identifié les similarités principales et les différences entre de nombreux modèles, puis nous avons abstrait des concepts pour définir des catégories d'entités, catégories que nous nommons axes. Chacun des axes possède des sous-axes, mais aussi une graduation. Par exemple, l'axe le plus important est celui du cycle de vie qui a 7 sous-axes : incrément, itération, parallélisme, gestion des retours arrière, durée du cycle, approche (descendante, ascendante, mixte . . .), et la focale (activités, produits, décision, contexte et but). Nous évaluons chacun des sous-axes pour caractériser les modèles de processus. Cette taxonomie est le résultat d'une étude bibliographique de plus de 150 articles qui a été évaluée par deux cas d'étude auprès d'étudiants de master. Ces études ont permis de montrer que ce cadre théorique leur permettait de comprendre de manière fine un modèle de processus, mais aussi de comparer plusieurs modèles de processus. Il existe donc une limitation évidente liée à la nature des sujets de l'expérimentation. Mais cela ne remet pas en cause l'idée qui ici n'est pas de définir des critères de qualité dans l'absolu, mais de définir les caractéristiques qui peuvent influer la qualité pragmatique d'un modèle de processus. L'hypothèse est qu'un modèle de processus n'est pas bon ou mauvais dans l'absolu, mais qu'il doit correspondre à une situation de conception. Aussi nous laissons au concepteur l'évaluation de la qualité du modèle tout en lui permettant de l'appréhender au mieux. Ce travail nous a montré que l'interprétation de la qualité doit parfois être laissée à l'appréciation du concepteur en fonction de son contexte de travail.

Du point de vue expérimental, nous avons abordé l'évaluation qualitative des modèles de processus. Dans (Dupuy-Chessa 2011), nous avons cherché à évaluer le modèle de processus d'une méthode de conception de système d'information ayant des interfaces homme-machine innovantes. L'étude théorique du modèle de processus n'étant pas suffisante pour juger de la convenance des activités, nous avons tenté de l'évaluer de deux manières pratiques différentes. Nous avons récolté des informations sur l'utilisation du modèle grâce à une approche qualitative, qui a pour but de parvenir à une compréhension fine du sujet étudié. Ainsi des échantillons petits mais ciblés sont utilisés. Les informations collectées sont nombreuses et souvent révélatrices de cas plus généraux même si leurs conclusions se limitent aux cas étudiés. Cette étude de cas a été menée auprès de 4 binômes de concepteurs (un spécialiste IHM et un spécialiste GL) pendant une semaine. Les concepteurs avaient pour consignes de suivre le processus proposé sur les phases les plus collaboratives pour concevoir un système interactif. Les résultats ont été positifs : le processus est perçu comme intéressant et satisfaisant (utile, permettant de réduire les erreurs, travail plus efficace). Toutefois la durée du projet et des échanges collaboratifs varie beaucoup d'un groupe à l'autre. En particulier, les collaborations avaient parfois lieu pour des objectifs inappropriés c'est-à-dire que des concepteurs travaillaient ensemble pour réaliser les modèles de l'un des domaines au lieu de travailler séparément. Ainsi l'étude de cas nous a amenés à simplifier le modèle de processus 1) pour ne conserver que les activités qui produisent des modèles indispensables à la suite du processus, les autres activités

devenant optionnelles ; 2) pour n'inclure que des coopérations dont le but est plus évident (la production d'un modèle commun). Si cette évaluation qualitative nous apportent des informations intéressantes sur le processus, elle n'a pas de valeur statistique et peuvent difficilement être reproduites. Une évaluation exhaustive du processus aurait nécessité de le comparer avec d'autres processus de développement. La mise en œuvre d'une telle expérience aurait toutefois été longue et complexe. Il est aussi difficilement envisageable d'adopter une approche quantitative qui nécessiterait la mise en place réelle du processus dans une perspective de fouille de processus qui n'apporterait pas d'explications sur les résultats observés. L'expérimentation qualitative nous semble être un outil indispensable pour améliorer a priori un modèle de processus.

Nous avons également travaillé sur la qualité des modèles de processus métiers (Cherfi et al., 2013). Du point de vue expérimental, nous avons collecté 100 modèles écrits en BPMN[1] depuis BPM Academic initiative[2]. Nous avons ensuite procédé à l'évaluation de la qualité de ces modèles en les confrontant à un ensemble de règles de correction syntaxique, sémantique et pragmatique permettant de détecter des défauts de qualité dans les modèles. La synthèse de cette évaluation révèle que 18% des défauts sont d'ordre syntaxique, 18% des défauts sont des erreurs liées à une mauvaise maîtrise de la sémantique des constructeurs de la notation et 64% des défauts altèrent la qualité pragmatique des modèles en rendant difficile leur compréhension. Ces résultats nécessitent une analyse plus approfondie pour essayer de comprendre les causes de ces défauts, leurs relations avec un enseignement et/ou une pratique de la modélisation des processus, le fait qu'ils aient été générés à partir de besoins détaillés et/ou formalisés etc. Une telle analyse n'a pas pu être menée puisque ces modèles ne sont pas associés à leur contexte de production et aucun élément sur le profil des personnes qui les ont produits n'est disponible. Une expérimentation contrôlée aurait pu être un bon moyen pour analyser toutes ces questions. Cependant, notre expérience passée concernant les expérimentations contrôlées a montré la difficulté d'avoir des échantillons suffisamment grand et variés permettant de généraliser les résultats obtenus.

Ces travaux nous permettent de noter la difficulté d'évaluer des modèles de processus (C8). L'approche théorique se limite à une caractérisation et l'approche expérimentale permet difficilement d'obtenir des résultats concluants : les évaluations qualitatives n'ont pas de valeur statistique et peuvent difficilement être reproduites. Il est aussi peu envisageable d'adopter une approche quantitative qui serait chronophage et pas nécessairement plus prolifique en retours d'utilisation. Combiner les deux approches (C9) en proposant de nouvelles pratiques expérimentales et de nouveaux outils est un point peu exploré qui semble être un besoin fort pour l'évaluation des modèles de processus.

[1] BPMN : Business Process Model and Notation : http://www.bpmn.org/
[2] http://bpmai.org/BPMAcademicInitiative/

4. Synthèse et analyse

Le Tableau 1 présente une synthèse de nos travaux présentés dans les sections précédentes. Pour cela nous nous sommes focalisés sur :

– l'objectif principal de l'expérimentation – nous en avons considéré deux types : évaluer la qualité elle-même, ou explorer les modèles pour la construction des nouvelles propositions et d'analyses plus générales sur l'évaluation des modèles ;

– l'approche utilisée – qualitative, quand les évaluations sont basées sur l'évaluation des évaluateurs ; ou quantitative, quand à l'inverse elles recourent aux métriques (mesures) collectées sur les modèles (avec des outils automatiques ou via des évaluations manuelles) ;

– les techniques d'évaluation – comme la révision (par exemple walkthrough et inspection), des enquêtes, ou des interviews directes avec les évaluateurs ;

– la nature des projets - projets de l'industrie ou académiques ; et,

– la décision finale sur la qualité des modèles - en considérant l'utilisation de seuils pré-definis, l'opinion d'experts ou les deux.

Ce tableau montre que l'évaluation de la qualité des modèles utilise toute la richesse des évaluations centrées utilisateurs : les expérimentations peuvent servir à évaluer, mais aussi à co-construire avec les utilisateurs ou à explorer de nouvelles solutions ; le qualitatif et le quantitatif sont deux techniques complémentaires qui visent à aborder des points de vue différents sur les modèles (le qualitatif permettant de recueillir des opinions et des idées, le quantitatif pouvant être utilisés de manière plus automatique grâce à la définition de seuil) ; une large palette de techniques d'évaluation centrée utilisateur (walkthrough, interview…) sont pertinentes pour évaluer la qualité des modèles ; l'opinion des experts semblent prépondérante.

A partir de cette synthèse et des constats que nous avons faits tout au long de l'article, nous pouvons en tirer les principales leçons suivantes :

– Quelque soit le modèle étudié, son évaluation n'est pas triviale. Dans tous les cas, il a été noté la nécessité de faire participer les différentes parties prenantes (C1, C3, C6), que ce soit pour évaluer le modèle en lui-même ou les métriques de mesure automatique de la qualité. L'approche centrée utilisateur est donc à explorer pour mieux les intégrer dans la conception et l'évaluation de modèles.

– Nous avons constaté que le langage de modélisation influence la qualité du modèle résultat (C2). Nous l'avons remarqué par exemple, pour un modèle de données dont l'évaluation est impactée par sa notation. Cette influence pourrait s'expliquer par la qualité de la notation (richesse, complexité, formalité etc.) ou par la maîtrise qu'ont les participants de cette notation. Aussi nous recommandons de créer une ingénierie des langages de modélisation qui ne se limite pas à des outils de création de modèles, mais permettent aussi d'en aborder la qualité du point de vue des concepteurs ou des lecteurs de ces modèles.

Référence	Objectif	Approche	Technique d'évaluation	Nature des projets	Décision finale
Mehmood et al. 2009	Explorer	Qualitative	Interviews	Académique	Basé sur l'opinion des experts
Si-said et al 2002	Évaluer	Quantitative	Semi-automatique via des métriques	Académique	
Akoka et al. 2008	Explorer et Évaluer	Quantitative et Qualitative	Enquête	Académique et industrie	Seuils et l'opinion des experts
Rabelo et al 1997	Évaluer	Qualitative et quantitative	Inspection	Académique	Seuils et l'opinion des experts
Ramos et al., 2004	Évaluer	Qualitative et quantitative	Walkthrough et outillage automatique	Industrie	Seuils et l'opinion des experts
Nery et al., 2006	Évaluation	Quantitative	Outillage automatique	Académique	Seuils
(Ceret et al 2013)	Evaluer	Qualitative	Outillage pour la comparaison	Académique	Opinion des experts
(Dupuy-Chessa 2011)	Explorer et Evaluer	Qualitative	Walkthrough et interviews	Académique	Opinion des experts

Tableau 1 – Synthèse et analyse des travaux

– L'importance de l'outillage a été notée (C4, C9). Elle a cependant bien été mise en relation avec les parties prenantes. Les calculs proposés, par exemple de métriques, doivent être conformes à ce qu'auraient réalisé des experts et doivent donc être validés par ces même experts, alors que les outils des définitions de syntaxes abstraite et concrète doivent être adaptées à leurs utilisateurs. Une vision centrée utilisateur est donc primordiale et doit être développée.

– Les expériences menées ont également mis en évidence un problème déjà soulevé dans la littérature qui est celui de la validation des approches de qualité proposées (C5).
– Dans le cas d'évaluations expérimentales, un expert est nécessaire pour l'interprétation (C7). Dans le cas des mesures, l'évaluation est simplifiée/automatisée, mais l'interprétation des résultats reste compliquée : il faut que les mesures s'accompagnent de seuils qui définissent la qualité des potentiels résultats. Ces seuils même s'ils sont validés, ne sont généralement pas suffisants et un expert reste primordial pour ne pas fournir une mauvaise interprétation des résultats. En effet, le contexte (taille du projet, caractéristiques de l'équipe de développement, caractéristique du deadline du projet, etc..) est déterminant mais est peu pris en compte, et,
– l'insuffisance d'outils automatiques pour l'évaluation, combinant si possible un ensemble de méthodes à ce sujet (C9), conduit à un processus d'évaluation long et couteux (C8). L'évaluation menée sur la qualité des processus métier montre surtout qu'un grand pourcentage des défauts de qualité constatés peut être évité avec un meilleur outillage aussi bien lors de la production de ces modèles que lors de l'évaluation de leur qualité. L'ingénierie des modèles devrait explorer plus avant les techniques d'évaluation afin d'obtenir plus aisément des indicateurs sur la qualité des modèles.

5. Conclusion et perspectives

La complexité croissante des systèmes à être développés pour répondre aux activités quotidiennes avec des technologies de plus en plus modernes ravive l'importance d'utiliser des modèles conceptuels avant l'implémentation des systèmes. Assurer la qualité de ces modèles est donc essentiel pour aboutir un système final qui soit réellement utilisé. Dans cet article, nous avons fait un recueil pour analyser les travaux que nous connaissons sur l'évaluation de la qualité des modèles. De manière générale nous avons remarqué 1) le besoin de l'implication de toutes les parties prenantes au sein du processus d'évaluation, 2) la nécessité d'utiliser des techniques bien établies et des outils automatiques pour aider dans le processus et 3) l'importance du langage de modélisation utilisé pour faciliter la compréhension et l'évaluation d'un modèle.

Les perspectives de ce travail sont donc de proposer une ingénierie des modèles et des langages centrée sur l'humain. On peut ainsi envisager des cycles de conception de langage où les futurs concepteurs, utilisateurs du langage, seraient parties prenantes dans la définition du langage et dans son évaluation ; des outils de métamodélisation qui ne restreindraient pas les pratiques des concepteurs de langages et leur permettraient de facilement le faire évaluer en fonction des besoins des concepteurs de modèles ; des outils de définition de métriques flexibles qui permettraient à chacun d'adapter ou de définir ses propres métriques sur les modèles etc. Les travaux que nous avons présentés dans cet article sont des premiers pas vers cette ingénierie centrée sur l'humain, qui, nous l'espérons, contribuera à l'obtention de modèles de meilleure qualité.

Remerciements

Les auteurs remercient l'association Inforsid pour son soutien financier pour l'organisation de réunions de travail. S. Dupuy-Chessa est aussi reconnaissante envers l'ANR pour son soutien sur ce sujet dans le cadre du projet MOANO.

Bibliographie

J.Akoka, I.Comyn-Wattiau , S.Cherfi, Si-said, 2008. Quality of conceptual schemas an experimental comparison, in: Second International Conference on Research Challenges in Information Science, 2008. RCIS 2008. Presented at the Second International Conference on Research Challenges in Information Science, 2008. RCIS 2008, pp. 197–208.

P. Assenova, P. Johannesson (1996), Improving quality in conceptual modelling by the use of schema transformations. In the proceeding of ER'96. Cottbus, Germany.

A. Bajaj (2002), Measuring the Effect of Number of Concepts on the Readability of Conceptual Models, In proceedings of the Workshop on the Evaluation of Modeling Methods in Systems Analysis and Design (EMMSAD) in conjunction with CAiSE, Toronto, Canada.

C. Batini, S. Ceri, S.B. Navathe (1992), Conceptual database design: An entity relationship approach, Benjamin Cummings, Redwood City, California.

S Becker, 2012. Model Transformations in Non-functional Analysis, in: Bernardo, M., Cortellessa, V., Pierantonio, A. (Eds.), Formal Methods for Model-Driven Engineering, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 263–289.

J. Cardoso, 2007. Business Process Quality Metrics: Log-based Complexity of Workflow Patterns, in: Proceedings of the 2007 OTM Confederated International Conference on On the Move to Meaningful Internet Systems: CoopIS, DOA, ODBASE, GADA, and IS - Volume Part I, OTM'07. Springer-Verlag, Berlin, Heidelberg, pp. 427–434

E. Céret, S. Dupuy-Chessa, Gaëlle Calvary, Agnès Front, Dominique Rieu (2013) A Taxonomy of Design Methods Process Models , Information and Software Technology, Elsevier, Volume 55, Issue 5, Pages 795–821.

S. Cherfi, Si-said, S. Ayad, , I. Comyn-Wattiau, , 2013. Improving Business Process Model Quality Using Domain Ontologies. J. Data Semant. 2, 75–87.

S. R. Chidamber, C.F. Kemerer (1994), A Metrics Suite for Object Oriented Design, IEEE Transaction Software Engineering, vol. 20 no. 6, p. 476-493

F.J. Domínguez-Mayo, Escalona M.J., Mejías M., and. Torres A. H 2010. A Quality Model in a Quality Evaluation Framework for MDWE Methodologies. In 2010 Fourth International Conference on Research Challenges in Information Science (RCIS) Pp. 495–506.

S. Dupuy-Chessa, Nadine Mandran, Guillaume Godet-Bar, et Dominique Rieu (2011) A case Study for Improving a Collaborative Design Process. Dans IFIP WG8.1 Working conférence on Method Engineering (ME'2011).

V. Guana, , D.Correal, 2011. Variability Quality Evaluation on Component-based Software Product Lines, in: Proceedings of the 15th International Software Product Line Conference, Volume 2, SPLC '11. ACM, New York, NY, USA, pp. 19:1–19:8.

K. Henricksen, J. Indulska (2004) Modelling and Using Imperfect Context Information, Proc the CoMeRea workshop at Percom'2004, Orlando, USA.

J. Krogstie, O. I. Lindland, G. Sindre (1995), Toward a deeper understanding of quality in requirements engineering, in: Proceedings of IFIP 8.1 Working Conference on Information Systems concepts (ISCO3): Towards a Consolidation of Views, pp. 82–95.

J. Krogstie, G. Sindre and J. Håvard (2006) : Process models representing knowledge for action: a revised quality framework. European Journal of Information Systems, vol 15, pp 91–102.

A. Levitin, T. Redman (1995), Quality dimensions of a conceptual view, Information Processing and Management, Vol 31(1).

O. Lindland, G. Sindre and A. Solvberg (1994). Understanding Quality in Conceptual Modeling. IEEE Softw. 11, 2, March 1994, 42-49.

M.V. Mantyla (2004) Developing new approaches for software design quality improvement based on subjective evaluations, Proceedings - International Conference on Software Engineering, vol. 26, p.48-50.

E. Marjomaa.(2002), Necessary Conditions for High Quality Conceptual Schemata: Two Wicked Problems, Journal of Conceptual Modeling, 27.

K.Mehmood, , S.Cherfi Si-said , , I.Comyn-Wattiau, , J. Akoka,., 2011. A pattern-oriented methodology for conceptual modeling evaluation and improvement, in: 2011 Fifth International Conference on Research Challenges in Information Science (RCIS). Presented at the 2011 Fifth International Conference on Research Challenges in Information Science (RCIS), pp. 1–11.

K. Mehmood, S. Si-Said Cherfi, I. Comyn-Wattiau (2009): Data Quality through Conceptual Model Quality - Reconciling Researchers and Practitioners through a Customizable Quality Model. ICIQ 2009: 61-74

D. L. Moody, G. G. Shanks (1994), What makes a good data Model? Evaluating the quality of entity relationship models, in: P. Loucopolous (Ed.), Proceedings of the 13th International Conference on the Entity Relationship Approach, Manchester, England, pp. 94–111.

P. Mohagheghi et J. Aagedal (2007) Evaluating quality in model-driven engineering. Dans Proceedings of the International Workshop on Modeling in Software Engineering, MISE '07, pages 6–, Washington, DC, USA, IEEE Computer Society.

P. Mohagheghi et V. Dehlen, A metamodel for specifying quality models in model-driven engineering, in; Proceedings of the Nordic Workshop on Model Driven Engineering, 2008, pp. 51–65.

A. Nery, L. Bandeira, F. Lima, K. M. Oliveira (2006) Qualidade de Software aplicada à navegabilidade na Web In: XII Simpósio Brasileiro de Sistemas Multimídia e Web.

A. Olivé (2007), Conceptual Modeling of Information Systems, Springer, Heidelberg.

JR A.Rabelo, A.R Rocha, K.M. Oliveira, A. Ximenes, A. Souza, C. Andrade, D. Onnis, N. Lobo, N. Ferreira, V. Werneck (1997) An Expert System for the Diagnosis of Acute

Myocardial Infarction with EKG Analysis. Artificial Intelligence in Medicine. , v.1, p.75 - 92.

C. Ramos, K.M. Oliveira, N. ANQUETIL (2004) Legacy Software Evaluation Model for Outsourced Maintainer, In: 8th European Conference on Software Maintenance and Reengineering, IEEE Computer Society. p.48 – 57

M. Rodriguez, M. Genero, D. Torre, B. Blasco, M. Piattini (2010) A methodology for continuos quality assessment of software artefacts, Proceedings - International Conference on Quality Software, p. 254-261.

P. Salvaneschi, U. Piazzalunga (2008) Engineering models and software quality models: An example and a discussion, Proceedings - International Conference on Software Engineering, p. 39-44.

S. Si-Saïd Cherfi, J. Akoka, I. Comyn-Wattiau (2002), Conceptual Modeling Quality - From EER to UML Schemas Evaluation, Proceedings of ER2002, Tampere (Finland)

B. Vanderose, N. Habra (2011) Tool-support for a model-centric quality assessment: QuaTALOG, 6th International Conference on Software Process and Product Measurement, p. 263-268.

Y. Wand, R. A. Weber (2002), Research commentary: information systems and conceptual modelling—a research agenda, Information Systems Research 13 (4) 363–376.

Vers une approche centrée humain pour la définition de langages de modélisation graphiques

Sophie Dupuy-Chessa[1], Benoît Combemale[2], Marie-Pierre Gervais[3], Thierry Nodenot[4], Xavier Le Pallec[5], Laurent Wouters[6]

1. Université de Grenoble Alpes, Laboratoire d'informatique de Grenoble
F-38000 Grenoble, France
Sophie.Dupuy@imag.fr
2. Inria et Université de Rennes 1, IRISA Campus de Beaulieu, 35042 Rennes Cedex benoit.combemale@irisa.fr
3. Laboratoire d'Informatique de Paris 6, Université Paris Ouest Nanterre La Défense,
4 Place Jussieu 75252 Paris Cedex 05marie-pierre.gervais@lip6.fr
4. Laboratoire d'Informatique de l'Université de Pau et des Pays de l'Adour,
IUT de Bayonne, 2 allée du Parc Montaury, 64600 Anglet
Thierry.Nodenot@iutbayonne.univ-pau.fr
5. Laboratoire d'Informatique Fondamentale de Lille
Bâtiment M3, Cité Scientifique, 59655 Villeneuve d'Ascq Cedex
xavier.le-pallec@univ-lille1.fr
6. CEA LIST
CEA Saclay Nano-INNOV 91191 Gif sur Yvettes
laurent.wouters@cea.fr

RESUME. *Avec la complexification des systèmes d'information (systèmes ubiquitaires, entreprises ouvertes, etc.), de nombreux nouveaux langages de modélisation sont proposés. Face à ce développement de langages spécifiques, leur qualité devient un enjeu important pour la modélisation des systèmes d'information. Cet article traite de ce problème en suggérant une approche centrée utilisateur pour la création de langages. Nos propositions s'appuient sur nos travaux antérieurs issus de différents projets, mais constituent un tout cohérent qui nous permet de montrer l'intérêt et la faisabilité de l'approche.*

ABSTRACT. *The increasing complexity of information systems (ubiquitous systems, open enterprises, etc.) calls for the introduction of always new modeling languages. However, the development of new domain-specific languages makes the question about their quality an important issue for the modeling of information systems. This article deals with this issue by suggesting a user-centred approach for language creation. Our proposals rely on our past work from different projects, but constitute a consistent whole, which allows us to show the approach interest and feasibility.*

MOTS-CLES : qualité, ingénierie des langages, définition centrée utilisateur

KEYWORDS: quality, modeling langage, user centred definition

1. Introduction

Les modèles, qui servent à comprendre et à représenter les systèmes, prennent une importance accrue face à la complexification des systèmes d'information (SI). Pour les gérer, l'ingénierie dirigée par les modèles (IDM) développe des techniques et des outils. Ainsi, il est maintenant possible de créer de nouveaux langages, spécifiques à un domaine (DSL). Ces nouveaux langages, s'ils ne sont pas définis et évalués avec le plus grand soin, risquent d'amener à la production de modèles inutiles ou inacceptables par les utilisateurs et donc à la remise en cause complète de ces langages de modélisation. Pour limiter les risques, il est nécessaire de s'intéresser à la manière dont ils sont créés. Dans cet article, nous nous intéressons à une approche centrée humain pour la définition des langages de modélisation en nous focalisant sur les langages graphiques. Nous basons notre réflexion sur l'étude des travaux existants en matière de qualité des langages et nous montrons comment, à travers différents projets, nous avons abordé la création de langages du point de vue de leurs concepteurs et de leurs futurs utilisateurs.

La section 2 présente un survol de la littérature existante sur la qualité des langages graphiques de modélisation. La section 3 décrit nos contributions pour la création de langages graphiques. Pour conclure, nous synthétisons en section 4 nos contributions et identifions quelques perspectives de ce travail.

2. Travaux existants

Dans cet article, nous utilisons une définition en intention d'un langage (description des propriétés communes aux instances possibles). Ainsi un langage est défini par une syntaxe abstraite, une syntaxe concrète et une sémantique. La syntaxe abstraite capture le vocabulaire et la taxonomie (i.e. les concepts) du langage (Fondement et Baar 2005) alors que la syntaxe concrète décrit la notation, c'est-à-dire la représentation des éléments du langage. En IDM, la syntaxe abstraite se décrit par un métamodèle ; la syntaxe concrète peut être graphique ou textuelle. Une séparation claire entre syntaxe abstraite et syntaxe concrète est une technique pour gérer la complexité de la définition d'un langage de modélisation car elle permet de définir les éléments d'un langage indépendamment de leur représentation. La description du langage est complétée par une sémantique. Nous n'aborderons pas ici la qualité de la sémantique, largement traitée par la thématique des langages formels et pour laquelle 4 formes de définition sont recommandées (Kleppe 2007) : dénotationnelle, opérationnelle, translationnelle et pragmatique.

Les travaux existants abordent souvent la qualité des langages de modélisation de manière globale. Dans ce cas, les expériences ou les cadres théoriques s'intéressent au langage indépendamment de ses éléments constitutifs (syntaxes abstraite, concrète et sémantique). Une autre approche consiste à étudier la qualité des composants du langage en se basant sur l'hypothèse que la qualité globale du langage est influencée par celle de ces composants. Ces deux approches sont présentées dans le survol de la bibliographie qui suit.

2.1. Cadre global de qualité

La qualité des langages est généralement assez difficile à appréhender. La plupart des travaux existants étudient un langage particulier en réalisant des expériences avec des utilisateurs. Ces évaluations ne sont néanmoins pas simples car il ne s'agit pas d'évaluer la qualité d'instances du langage (par exemple, un diagramme de classes pour un système bancaire), mais celle du langage (par exemple, le diagramme de classes en UML). Ainsi Siau et Tian (Siau et Tian 2001) utilisent une approche basée sur le modèle de traitement de l'information GOMS (Goals Operators Methods and Selection Rules (Card et al. 2000)) pour évaluer des diagrammes UML. Les auteurs mesurent le temps d'exécution pour réaliser certains des diagrammes UML et déterminer leur complexité. De manière plus générique, (Aranda et al. 2007, Patig 2008) proposent des protocoles expérimentaux applicables à n'importe quel langage pour évaluer leur facilité de compréhension.

De manière complémentaire à ces travaux, des référentiels étudient les langages dans leur globalité (syntaxes et sémantique). Ils s'inscrivent dans l'approche sémiotique pour la qualité des modèles. Ainsi (Krogstie 2003) identifie 5 caractéristiques pour un langage de modélisation : 1) l'adéquation au domaine ; 2) l'adéquation aux connaissances des acteurs (i.e. les concepteurs de modèles) ; 3) l'adéquation à la capacité d'externaliser les connaissances : il ne doit pas y avoir d'assertions dans les connaissances explicites des participants qui ne peuvent être exprimées dans le langage. ; 4) l'adéquation à la capacité de compréhension des parties prenantes; et 5) l'adéquation à l'interprétation par des acteurs techniques (les outils de modélisation, de simulation, de preuve etc).

Les travaux que nous venons de citer apportent une caractérisation des langages. Pour autant, la qualité peut s'aborder sous un angle différent, qui est celui des différents composants des langages, restreints dans cet article et comme mentionné dans les sections précédentes à la syntaxe abstraite et la syntaxe concrète graphique. Etudier la qualité selon ces deux aspects permet de cibler au mieux les questions de recherche ouvertes.

2.2. Qualité de la syntaxe abstraite

La qualité de la syntaxe abstraite est relative au métamodèle et se base sur son évaluation. Ainsi les travaux décrits ci-dessous ont cherché à établir un lien entre la qualité du métamodèle et celle du langage.

Dans (Mohagheghi et Aagedal 2007), l'hypothèse est qu'un métamodèle complexe conduit à un pouvoir d'expression plus grand et donc à des modèles plus petits. Suivant la même approche, (Rossi et Brinkkemper 1996) propose une méthode de calcul de la complexité conceptuelle théorique. Dans ce cas, l'hypothèse (non démontrée) est qu'il existe une dépendance intrinsèque entre les métamodèles et la facilité d'apprentissage du langage : « the more complex a metamodel, the harder the method is to learn ». Ce travail a permis de comparer plusieurs langages de modélisation orientés objet à partir de leur métamodèle et conclut qu'ils

deviennent plus complexes au fil du temps. En utilisant les mêmes règles de calcul, Siau et Cao cité dans (Krogstie 2003), aboutissent à des résultats similaires : UML est de 2 à 11 fois plus complexe que n'importe quel autre langage de modélisation orienté-objet.

2.3. Qualité de la syntaxe concrète graphique

Comme introduit précédemment, la syntaxe concrète d'un langage de modélisation peut être textuelle ou graphique. Dans le contexte de cet article, nous nous concentrerons uniquement sur les syntaxes graphiques qui sont utilisées par les concepteurs de modèles conceptuels.

Les travaux étudiant la qualité de la syntaxe concrète s'intègrent dans l'approche de la sémiologie graphique. Ils considèrent que l'objectif d'un modèle est de transcrire graphiquement une information. Dans ce cadre, nous pouvons citer les travaux de Bertin (Bertin 1967) qui propose d'exploiter les capacités du système visuel humain à percevoir la profondeur des objets que nous voyons. Par exemple, la taille a un fort pouvoir d'ordonnancement (profondeur donc) : dans un diagramme de classes, l'augmentation significative de la taille de la police de caractères utilisée pour l'affichage du nom des paquetages, permettrait de mettre en avant les paquetages présents.

Par la suite, le principal travail pour appréhender la qualité d'une notation graphique est la Théorie de la Physique des notations (Moody 2009) qui donne 9 principes pour concevoir des notations visuelles cognitivement efficaces. Nous citerons à titre d'exemples la transparence sémantique, qui traite en partie du lien entre syntaxe concrète et sémantique, et la discriminabilité perceptive qui est entièrement focalisée sur la notation :

- La transparence sémantique définit dans quelle mesure la signification d'un symbole peut être déduite de son apparence. Les symboles doivent donc fournir des indices sur leur sens (la forme implique le contenu). Ce concept est proche de celui d' "affordance" en interaction homme-machine : l'"affordance" cherche la transparence dans les actions possibles pour l'utilisateur alors que la transparence sémantique vise la facilité de compréhension des concepts. La transparence sémantique n'est pas un état binaire, mais un continuum allant de la compréhension immédiate de la signification du symbole sans explication jusqu'à une interprétation différente ou opposée à son sens.
- La discriminabilité perceptive. La première phase de traitement de l'information chez l'être humain est la perception sensorielle. Pour un schéma graphique, cette perception est visuelle : reconnaissance des formes, des couleurs... Cette activité bénéficie d'une puissance de calcul importante (une partie du cerveau dédiée représentant plus d'un quart du cerveau) grâce notamment à un fonctionnement massivement parallèle. Le fait de pouvoir discerner facilement chaque type d'élément graphique par rapport aux autres est donc primordial. Cette propriété se nomme la discriminabilité perceptive.

Si on se réfère à la théorie de compréhension des graphes de Pinker (Pinker 1990), les critères d'évaluation des notations visuelles (en Ingénierie Logicielle) de Moody interviennent soit au niveau perceptif (p. ex., discriminabilité perceptuelle), soit au niveau cognitif (p. ex., transparence sémantique, clarté sémiotique) soit aux deux (p. ex., gestion de la complexité). L'intérêt de cette théorie, qui synthétise de nombreux travaux en visualisation, est d'indiquer que des mécanismes liés à ces critères sont «innés» et d'autres sont fonction de l'histoire du lecteur.

Comme expliqué dans (Moody 2009), les travaux sur la physique des notations sont complémentaires des travaux plus anciens de T. Green qui portent non pas sur une évaluation détaillée des seules notations visuelles mais plutôt sur des critères d'analyse de l'utilisabilité des environnements visuels exploitant de telles notations (Green & Petre 1996), (Green & Blackwell 2003), (Green et all 2006). Ainsi, les travaux de Green proposent quatorze dimensions d'analyse que des non spécialistes peuvent percevoir et discuter lorsqu'ils utilisent les environnements visuels mis à leur disposition. Certains critères de D. Moody et T. Green se recoupent cependant (la dimension « Abstraction » de T. Green est un élément important du principe de « Gestion de la complexité » de Moody ; la dimension « Dépendance cachée » de T. Green est à rapprocher du principe de « Transparence sémantique » de D. Moody, etc.). Mais le recouvrement n'est que partiel car pour T. Green, la notation visuelle n'est que l'un des éléments d'un puzzle visant à produire des instruments adaptés à l'activité et aux spécificités des usagers (Fig. 1) : notation et environnement doivent être intégrés comme le montre la partie droite de la figure et non juxtaposés.

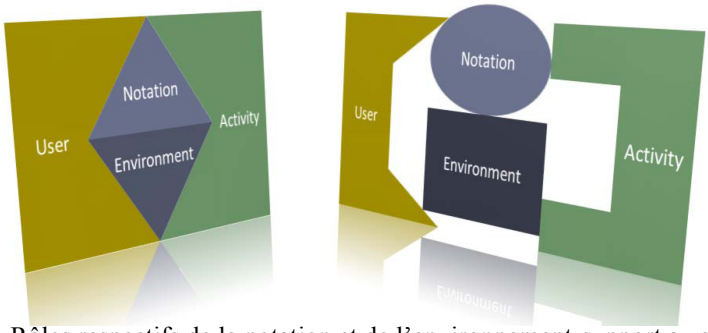


Figure 1. Rôles respectifs de la notation et de l'environnement-support au service de l'activité de modélisation.

Aussi, par la suite, nous nous appuierons également sur trois dimensions des travaux de T. Green qui nous paraissent influer fortement sur la capacité qu'aura un utilisateur à explorer ou rechercher, encoder, modifier des éléments dans un modèle via un environnement visuel :

- La visibilité : la capacité de la notation et de l'environnement à rendre compte des différentes facettes des modèles.
- L'évaluation progressive : la capacité de la notation et de l'environnement à supporter une évaluation progressive des modèles produits.

- L'engagement prématuré : capacité à éviter au concepteur de devoir prendre des décisions de modélisation prématurées compte tenu de son rôle et de l'objectif de son activité.

2.4. Synthèse

Au travers de notre revue de la littérature sur la qualité des langages (syntaxe abstraite et syntaxe concrète graphique), nous pouvons observer que le concepteur d'un langage de modélisation a de nombreuses contraintes à prendre en compte : l'adéquation au domaine, à la connaissance des acteurs, s'adapter aux limitations de l'éditeur potentiel... En plus de ces contraintes générales, il ou elle doit aussi prendre garde à ne pas rendre trop complexe la syntaxe abstraite au fil du temps et être au fait de l'importance de la syntaxe concrète. Malheureusement, le point de vue du concepteur désireux de créer un langage de modélisation de qualité n'est pas pris en compte par les précédents travaux. Ainsi la question de la disponibilité de techniques et outils permettant aux concepteurs de langages de modélisation d'être performants et efficaces reste ouverte. C'est un point que nous aborderons dans la section suivante.

3. Vers une approche centrée humain pour la création de langages

Nous abordons la qualité des langages de modélisation en considérant les deux facettes, syntaxes abstraite (partie 3.1) et concrète (partie 3.2), ainsi que les liens qu'elles peuvent avoir (partie 3.3). Cette section présente des travaux issus de différents projets, non coordonnés, auxquels les auteurs ont pu participer. Ils ont néanmoins tous en commun d'avoir suivi une approche centrée sur le point de vue de l'humain pour la définition de nouveaux langages.

3.1. Définition de la syntaxe abstraite

La qualité de la syntaxe abstraite est relative au métamodèle. Pour le définir, les experts qui souhaitent capturer précisément le périmètre de leur domaine doivent maîtriser deux formalismes différents: un langage orienté-objet aligné sur MOF pour modéliser la structure du domaine; et un langage supportant la logique du premier ordre aligné sur OCL pour ajouter les contraintes nécessaires précisant la structure des modèles attendus. L'utilisation conjointe des deux langages de métamodélisation représente un défi majeur qui n'est à ce jour pas supporté par des méthodologies ou des bonnes pratiques. Ce dernier point est particulièrement difficile dans le cas de l'évolution de l'une ou l'autre des vues. Un cas notable de l'évolution de la norme UML à l'OMG est la classe *AssociationEnd* qui a disparu du métamodèle après la version 1.4 en 2003. Pourtant, jusqu'à la version 2.2, sorti en 2009, il a subsisté des expressions OCL se référant à cette classe (Selic 2008). De la même manière, la spécification d'OCL 2.2 dépend de MOF 2.0, mais une section particulière de la spécification définissant la liaison entre MOF et OCL (OCL2, p.169, 2010) fait usage de la classe *ModelElement* qui n'a existé que jusqu'à la version 1.4 de MOF.

Pour comprendre et essayer d'améliorer l'utilisation conjointe de ces deux formalismes, nous avons mené une étude empirique sur 1262 contraintes issues de 33 métamodèles (Cadavid et al, 2012). Cette étude nous a amené à définir formellement de nouvelles métriques (p.ex., taille du métamodèle, nombre d'invariant (total et par contexte), complexité des invariants par rapport au métamodèle, complexité des invariants par rapport au langage OCL, etc.) dédiées à l'évaluation de la qualité d'un métamodèle décrit en utilisant conjointement MOF et OCL, et permettent ainsi de révéler différents aspects du couplage et de la dispersion. L'ensemble des métriques a été intégré dans un outil permettant l'analyse automatique d'un métamodèle fourni en entrée. Nous observons sur les données prises en compte dans l'étude empirique que les experts tendent vers une utilisation dans leurs contraintes d'un petit ensemble des concepts de leur domaine (i.e., la majorité des contraintes utilise moins de 5 concepts du domaine). Malgré ce faible couplage, nous observons également que l'utilisation conjointe de MOF et OCL soulève des problèmes de consistance. 422 contraintes parmi les 1262 n'ont pas pu être analysées à cause d'un problème de lien avec la structure du métamodèle décrit à l'aide de MOF. Bien que OCL soit devenu un standard de facto pour la définition de contraintes sur une structure orientée-objet décrite à l'aide de MOF, ce n'est pas son objectif initial. En conséquence, nous observons également qu'une partie très significative du langage n'est jamais utilisée dans des contraintes d'un métamodèle : 10 sur les 22 concepts d'OCL ne sont jamais utilisés dans notre ensemble de données. Cette étude confirme la difficulté pour un expert d'un domaine à utiliser conjointement MOF et OCL.

Capturer précisément un domaine au sein d'un langage de modélisation assure la pertinence des modèles construits à partir de ce langage vis-à-vis du domaine considéré. Néanmoins nous avons pu constater la difficulté pour un expert d'un domaine à définir une syntaxe abstraite avec les techniques de l'IDM (MOF et OCL). Une perspective immédiate est donc d'offrir une base de recommandations et un outillage aidant les experts à utiliser conjointement MOF et OCL pour capturer précisément leur domaine dans un métamodèle.

3.2. Définition de la syntaxe concrète graphique

La syntaxe concrète graphique s'intéresse à définir des représentations adéquates pour les concepteurs. Nous avons d'une part, cherché à mesurer certains critères de qualité lors de la création de la syntaxe concrète graphique ; d'autre part, nous étudions le caractère social d'une notation c'est-à-dire la capacité d'une notation à permettre à des acteurs de la modélisation de travailler ensemble.

Pour mesurer la qualité de la syntaxe concrète graphique, nous cherchons à identifier des métriques qui aideront les concepteurs de langages à penser et à évaluer leur syntaxe. ModX[1], un méta-éditeur existant développé depuis plusieurs années au LIFL (et présenté dans la partie 3.3) a été étendu pour ajouter des

[1] Site web de ModX, http://www.lifl.fr/modx

métriques calculables sur toute syntaxe concrète graphique définie dans ModX (Le Pallec et Dupuy-Chessa 2011) (le Pallec et Dupuy-Chessa 2012). Les principales propriétés «visuelles» identifiables dans la physique des notations ont été inventoriées (Le Pallec et Dupuy-Chessa 2012). Ce travail nous montre qu'il est très difficile, voire impossible pour certains critères, de définir des mesures automatiques de qualité (p.ex., transparence sémantique, adaptation cognitive, gestion de la complexité). Les métriques ne peuvent donc être envisagées que comme des indicateurs partiels et doivent forcément être couplées avec des évaluations expérimentales. Elles peuvent néanmoins aider les concepteurs de langages à proposer une syntaxe concrète de meilleure qualité.

La syntaxe concrète d'un langage de modélisation peut aussi s'analyser dans ce qu'elle entretient avec les acteurs qui la manipulent au service d'une tâche donnée, dans les ponts qu'elle permet d'établir entre des acteurs différents tant dans les objectifs que dans les compétences en modélisation. Nous avons acquis l'expérience suivante lors de la définition et de la mise en œuvre de deux langages graphiques de modélisation :

- le langage offert par l'environnement CPM (Laforcade, Nodenot et al. 2005), (Nodenot, Laforcade et al. 2007), (Nodenot 2008)
- et le langage offert par l'environnement WindMash (Luong et al. 2011), (Luong et al. 2012)

Ces langages nous ont ainsi conduit à considérer que la notation visuelle et plus généralement que l'environnement support à l'exploitation d'une notation visuelle sont des éléments clés pour articuler les perspectives de modélisation de différents acteurs appartenant à des mondes sociaux différents mais travaillant sur un projet commun.

Dans ces travaux, il s'agissait d'amener des enseignants et des informaticiens à se coordonner malgré leurs points de vue différents, de leur permettre de construire des compréhensions communes sans perdre la diversité de leurs points de vue. Lors de nos expérimentations, nous avons pu constater qu'un effort tout particulier est nécessaire pour définir les notations mises à disposition de ces acteurs afin de leur permettre de décrire/manipuler/interroger des objets frontières de modélisation que S.L. Star définit comme : des objets abstraits ou concrets dont la structure est suffisamment commune à plusieurs mondes sociaux pour qu'elle assure un minimum d'identité au niveau de l'intersection tout en étant suffisamment souple pour s'adapter aux besoins et contraintes spécifiques de ces mondes (Star 1989), (Star 2010). Supportée par la notation, la flexibilité d'interprétation d'un objet-frontière peut alors devenir le support à des traductions hétérogènes, comme dispositif d'intégration des savoirs, comme médiation dans le processus de coordination d'experts et de non experts (Trompette et Vinck 2009).

Les retours d'expériences nous ont amenés aux conclusions suivantes sur les capacités que doivent avoir les notations et l'environnement de modélisation exploitant ces notations complémentaires des objets-frontières.

Concernant la capacité à rendre compte des différentes facettes des modèles (cf. Visibilité), l'environnement-support doit faciliter la juxtaposition des représentations complémentaires d'un objet-frontière. D'une part, les relations qu'entretiennent les

diverses notations d'un même concept doivent être explicitées lors de la définition des syntaxes concrètes mais l'environnement de modélisation doit en plus concrétiser matériellement ces relations pour les utilisateurs lorsque les deux représentations sont visibles et juxtaposées. Nous avons ainsi dû intégrer dans l'environnement WindMash des mécanismes de notification spécifiques que chaque éditeur de modèles devait exploiter pour concrétiser les liens entre objets frontières : contenus à valoriser rendus disponibles au niveau de la conception des interfaces-usager via des opérations de drag'n drop, composants d'interface devenant des lignes de vie lors de la spécification des interactions possibles entre un utilisateur et ces composants d'interface (voir *http://youtu.be/3uxR8euHPwM?hd=1*)**.**

La syntaxe concrète doit supporter une évaluation progressive des modèles produits (cf. Evaluation Progressive). Les objets-frontières représentant des enjeux bien particuliers dans le processus de modélisation, il est essentiel que la syntaxe concrète des langages facilite l'évaluation des modèles par les concepteurs au fur et à mesure de l'avancement de l'activité. Dans le cas de l'environnement WindMash et des Live Sequence Charts, des éléments de la notation vont porter les résultats de l'exécution partielle d'un modèle, facilitant ainsi l'évaluation de la pertinence du travail de conception d'un objet-frontière donné. Si l'exécutabilité des modèles est bien sûr un prérequis des environnements-support correspondants, il est important de noter que l'environnement exécutant ces modèles doit pouvoir modifier à la volée/valoriser/visualiser les éléments dont l'état a changé, la dynamique du cycle modélisation/exécution des modèles étant donc portée par la notation.

Enfin il est important d'éviter au concepteur de devoir prendre des décisions de modélisation prématurées compte tenu de son rôle et de l'objectif de son activité (cf. Engagement Prématuré). Pour pouvoir exécuter des modèles, un environnement-support doit s'appuyer sur des modèles totalement définis, condition rarement satisfaite surtout lorsque les concepteurs ne sont pas des spécialistes en modélisation mais des spécialistes de leur domaine (par exemple des enseignants amenés à coopérer avec des informaticiens pour concevoir des applications éducatives). Plutôt que de forcer les concepteurs à prendre des décisions dont ils ne perçoivent ni les conséquences ni l'intérêt, les travaux que nous venons de décrire ont montré qu'il était efficace d'affecter des valeurs par défaut aux éléments de modèles non précisés explicitement par le concepteur à condition que ces éléments portant des valeurs par défaut soient modifiables via le système de notation proposé au concepteur. C'est donc au cours de l'exécution de modèles que les acteurs pourront découvrir, par l'expérience, ces éléments de modélisation complémentaire : la notation devient alors le support à un processus d'ajustement entre le comportement perçu du modèle et le comportement à atteindre construit par un processus de résolution de problème.

Qu'il s'agisse de la définition des caractéristiques perceptives ou des caractéristiques cognitives d'une notation graphique, nous retenons le rôle des environnements-supports (notamment des méta-éditeurs) pour mesurer la qualité des notations et procéder aux nécessaires ajustements requis pour satisfaire aux différents critères de qualité énoncés. Ces ajustements concernent tout autant la notation proprement dite que l'articulation entre notations comme moyen d'établir des ponts entre les préoccupations de différents acteurs.

3.3. Influence entre syntaxes concrète et abstraite

La définition d'un langage de modélisation dédié passe par l'élaboration de sa syntaxe abstraite et de sa syntaxe concrète. Quelles sont les dépendances entre ces deux types de syntaxes, quelles sont leurs mises en relation et quelle est la répercussion de l'une sur l'autre ? Cette section étudie l'impact de cette mise en relation sur une approche orientée humain de la construction de langages de modélisation et des éditeurs les supportant.

Les artefacts intervenants dans cette production sont présentés sur la figure 2. Un éditeur est généralement construit à partir i) du modèle du domaine métier (c'est-à-dire le métamodèle représentant la syntaxe abstraite), ii) d'un modèle de la syntaxe concrète (décrivant par exemple des représentations textuelles ou graphiques) et iii) d'un modèle de mapping permettant de configurer le lien entre les deux modèles précédents.

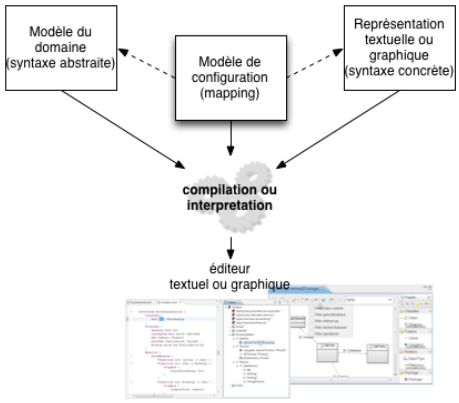


Figure 2. Définition d'un éditeur dédié à un domaine.

La méthodologie traditionnellement (Ráth et al. 2010) utilisée dans la communauté IDM pour produire un tel éditeur, que nous appellerons par la suite « approche métamodèle », est la suivante :

1. Eliciter les concepts du domaine et construire la syntaxe abstraite du langage (le métamodèle). Cette étape peut être réalisée par des entretiens avec les experts du domaine pour mettre en lumière les concepts du domaine et leurs relations ;
2. Produire la syntaxe concrète du langage par l'association de symboles aux éléments de la syntaxe abstraite. Ces symboles peuvent être visuels dans le cas d'un langage de modélisation graphique, ou textuel si un langage textuel est recherché.

Cette méthodologie peut être simplifiée pour le concepteur du langage par l'utilisation d'outils dédiés à cela, et généralement inclus dans des outils appelés « meta-tools » (Ráth et al. 2010), (Grundy 2008).

La caractéristique de cette méthodologie est qu'elle met l'accent sur la syntaxe abstraite du langage, ce qui peut être un avantage ou un inconvénient suivant les cas. Ce focus sur la syntaxe abstraite permet de répondre au mieux aux attentes en terme de manipulation des données qui seront décrites dans ce langage (requêtes, transformations, etc.). En revanche, la syntaxe concrète est le parent pauvre de cette approche et peut être limitée par les choix faits lors de la production de la syntaxe abstraite. Evans et al. (Evans et al., 2009) montrent que les concepteurs sont peu enclins à apporter des modifications au métamodèle même lorsque la construction de la syntaxe concrète amène à sa remise en cause. C'est alors la syntaxe concrète qui est adaptée pour correspondre au métamodèle, parfois au détriment des concepteurs. Or ce sont eux qui manipulent la syntaxe concrète. Une autre limite de cette approche est le couplage fort entre les deux syntaxes, qui ne permet qu'une simple mise en correspondance entre les deux syntaxes, comme illustré dans (Ráth et al. 2010).

Aussi Wouters et al. explorent une approche inverse, appelée « approche notationnelle » et mettant l'accent sur la syntaxe concrète dans le cas de langages graphiques (Wouters et al., 2013) :

1. Produire la syntaxe concrète en essayant de respecter au mieux les notations existantes dans le domaine visé ;
2. Déduire la syntaxe abstraite de la syntaxe concrète et l'adapter au besoin.

Cette approche notationnelle a pour objectif de produire un langage le plus proche possible des attentes des experts en termes de notation. Elle est particulièrement applicable lorsqu'une notation préexiste dans le domaine et qu'il est important de reprendre le plus possible celle-ci afin de maximiser l'adoption du langage par les experts.

La comparaison de ces deux approches méthodologiques a fait l'objet d'une étude empirique avec des étudiants de Master 2 (Wouters et al., 2013). Les résultats obtenus par comparaison sur un cas commun montrent que l'approche notationnelle procure une meilleure proximité avec les attentes des experts du domaine. De même, les syntaxes abstraites obtenues par l'approche notationnelle sont de meilleure qualité. En outre, cette étude montre qu'il existe une forte corrélation entre la proximité de la syntaxe concrète et la qualité de la syntaxe abstraite avec l'approche notationnelle. Ceci s'explique par le fait que la syntaxe abstraite est déduite de la syntaxe concrète. Cette corrélation est une propriété importante de l'approche notationnelle, car elle montre que la construction de syntaxes concrètes de meilleure qualité conduit à des syntaxes abstraites également de meilleure qualité.

L'approche notationnelle démontre l'importance de la prise en compte de la syntaxe concrète dans la création d'éditeurs de modèles. Cette prise de conscience est similaire à celle, il y a de nombreuses années, concernant l'importance de l'interaction Homme-Machine lors de la conception d'applications informatiques. Profiter des bonnes pratiques en ingénierie logicielle pour la création d'éditeurs de modèles a été notre objectif lors de la création du «meta-outil ModX. Nous sommes donc partis d'une réflexion très générale sur la conception d'application, pour

déterminer quelles caractéristiques devait présenter notre outil afin de supporter toute démarche efficace de création d'éditeurs de modèles.

Concevoir un éditeur de modèles est aussi concevoir une application informatique avec une partie fonctionnelle (la syntaxe abstraite), une partie non-fonctionnelle (essentiellement la persistance) et une partie interactive (la syntaxe concrète et l'outillage associé). Ainsi avec l'outil ModX, nous avons voulu supporter, d'un point de vue logiciel, toute démarche liée à la conception d'éditeurs de modèles où le prototypage serait une nécessité. Pour cela, le cahier des charges de ModX était de permettre à un concepteur (de métamodèles) de pouvoir changer à tout moment d'aspect, c'est-à-dire définition de la syntaxe abstraite, définition de la syntaxe concrète et de l'outillage, et édition de modèles. Tous s'exécutent en parallèle et ModX assure continuellement la cohérence entre chacun de ces aspects.

Le premier avantage de notre solution est de permettre de tester rapidement les capacités d'expression d'un métamodèle nouvellement créé car un premier éditeur est non seulement fourni instantanément mais est facilement personnalisable. Enfin, grâce aux propriétés réflexives de ModX, chacun des 3 aspects pilotant un éditeur de modèles peut être modifié à tout moment, et les répercutions seront automatiques sur les modèles en cours d'édition. C'est ce que montre la figure 3 : si le ou les modèles créés ne conviennent pas, la possibilité de modifier un ou des aspects du langage de modélisation et de voir les répercussions dynamiquement permet de déterminer plus vite les évolutions nécessaires à apporter.

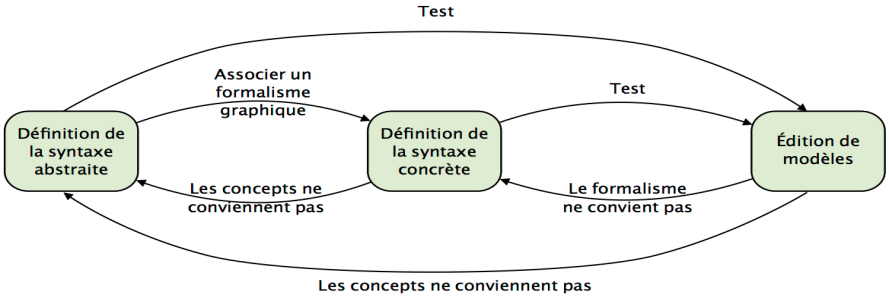


Figure 3. Cycle de vie classique de ModX

Suivant la même approche notationnelle, (Mandran 2013) définit un autre processus de création de langage où l'évaluation est centrale (Fig. 4). La définition de ce processus est le résultat d'une demi-douzaine d'expérimentations faites au Laboratoire d'Informatique de Grenoble lors de la création de langages. Les deux principes qui y sont défendus sont que d'une part l'évaluation d'un langage doit être réalisée tout au long du processus de développement d'un langage (i.e. lors de ses phases d'analyse, de conception et d'opérationnalisation) et que d'autre part l'évaluation doit être réalisée par des expérimentations centrées utilisateur. Il s'agit d'impliquer les futurs utilisateurs d'un langage dès l'étape d'analyse du futur langage. Des expérimentations impliquant les futurs utilisateurs du langage doivent être menées tout au long du processus de développement, certes pour valider le

langage, mais également pour l'explorer et le co-construire (cycles exploration/co-construction/validation sur le bas de la figure).

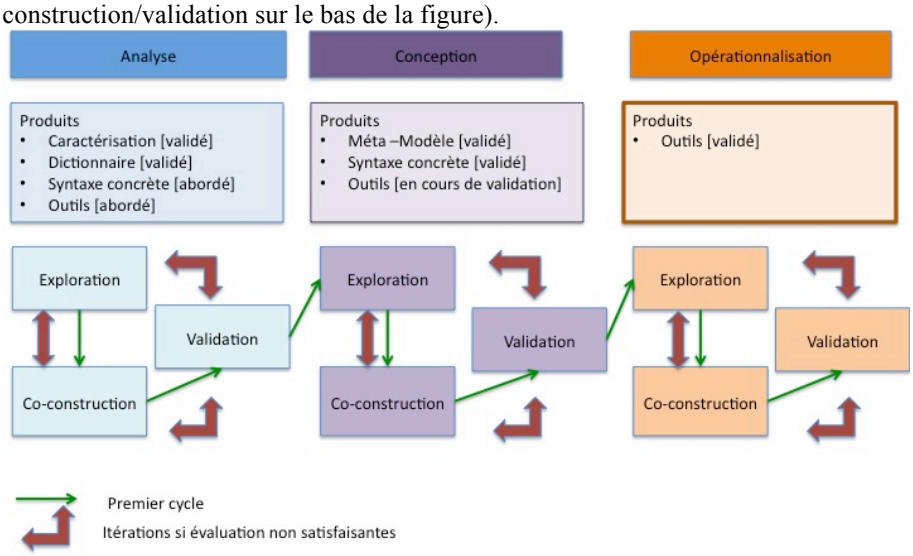


Figure 4. Cycle d'évaluation intégré au cycle de développement d'un langage

Les approches notationnelle et itérative remettent en question la primauté et la prédominance de la syntaxe abstraite lors de la conception d'un éditeur de modèles. Syntaxes abstraite et concrète sont deux facettes d'un même système, et la partie abstraite n'a pas à être favorisée. L'approche notationnelle va même jusqu'à privilégier la syntaxe concrète. Si on considère que l'aspect interactif est une préoccupation majeure de cette syntaxe, privilégier celle-ci rappelle les démarches de conception centrées sur l'humain où l'utilisateur/l'utilisation est centrale. La conférence Models a d'ailleurs récompensé les travaux de (de Lara 2012) qui préconise une approche ascendante où la définition d'un métamodèle est principalement faite au travers d'exemples d'utilisation.

4. Conclusion et perspectives

Cet article nous a permis de montrer l'intérêt d'une approche centrée sur l'humain pour la définition de langages de modélisation dédiés. Nos travaux sont le fruit d'une analyse conjointe a posteriori à partir des données d'usage que chacun des auteurs a pu recueillir à l'occasion de projets de longue durée mettant des utilisateurs en situation de modéliser à l'aide d'un langage dédié. La principale leçon qui ressort de cette mise en commun des expériences porte sur la nécessaire intégration des utilisateurs d'un langage durant sa conception que ce soit pour la définition et l'évaluation des éléments du langage. En particulier, l'outillage doit permettre de définir de manière aisée la syntaxe abstraite, en prenant en compte les difficultés liées à la manipulation du MOF et d'OCL ; il doit aider à mesurer la qualité des notations, tout en permettant de nécessaires ajustements pour mieux prendre en compte les utilisateurs ; et il doit apporter de la flexibilité dans la création

du langage pour supporter une approche centrée sur les concepteurs et les utilisateurs des langages.

Les perspectives de ce travail sont donc de proposer une ingénierie des langages centrée sur l'humain. On peut ainsi envisager des cycles de conception flexibles de langage où les futurs concepteurs, utilisateurs du langage, seraient parties prenantes dans la définition du langage et dans son évaluation ; des outils de métamodélisation qui ne restreindraient pas les pratiques des concepteurs de langages et leur permettraient de facilement le faire évaluer en fonction des besoins des concepteurs de modèles ; des outils de définition de métriques flexibles qui permettraient à chacun d'adapter ou de définir ses propres métriques sur les langages etc. Ainsi nous devons adapter les solutions proposées en ingénierie de la conception de système au domaine de la conception de langage de modélisation graphique. Les travaux que nous avons présentés dans cet article sont des premiers pas vers cette ingénierie, qui prend le parti d'une approche centrée sur l'humain, qui, nous l'espérons, contribuera à l'obtention de modèles et de langages de meilleure qualité car tout autant expressifs et mieux acceptés.

Remerciements

Les auteurs remercient l'association Inforsid pour son soutien financier pour l'organisation de réunions de travail. S. Dupuy-Chessa, X. Le Pallec et T. Nodenot sont aussi reconnaissants envers l'ANR pour son soutien sur ce sujet dans le cadre du projet MOANO.

Bibliographie

J. Aranda, N. Ernst, J. Horkoff, et Steve Easterbrook (2007) A framework for empirical evaluation of model comprehensibility. Dans Proceedings of the International Workshop on Modeling in Software Engineering, MISE '07, pages 7–, Washington, DC, USA, IEEE Computer Society.

J. Bertin (1967). Sémiologie graphique: les diagrammes, les réseaux, les cartes. The Hague, Paris: Mouton and Gauthiers-Villars.

A. Blackwell, and T. Green. (2003) Notational systems - the Cognitive Dimensions of Notations framework. Dans J.M. Carroll (Ed.) HCI Models, Theories and Frameworks: Toward a multidisciplinary science. San Francisco: Morgan Kaufmann, pages 103-134

J. Cadavid, B. Combemale, and B. Baudry (2012) Ten years of Meta-Object Facility: an Analysis of Metamodeling Practices, INRIA, Rapport de recherche RR-7882.

S. K. Card, A. Newell, and T. P. Moran (2000) The Psychology of Human-Computer Interaction. L. Erlbaum Associates Inc., Hillsdale, NJ, USA.

J. de Lara, J. Sánchez Cuadrado and E. Guerra. (2012) Bottom-up meta-modelling: An interactive approach. Lecture Notes in Computer Science 7590, Springer. pp.: 3-19. Presented at MODELS'12: ACM/IEEE 15th International Conference on Model Driven Engineering Languages and Systems. Springer best paper award.

A. Evans, M. Fernandez, and P. Mohagheghi (2009) Experiences of Developing a Network Modeling Tool Using the Eclipse Environment. In Model Driven Architecture -

Foundations and Applications, volume 5562 of Lecture Notes in Computer Science, pages 301-312. Springer Berlin / Heidelberg.

F. Fondement and T. Baar. (2005) Making metamodels aware of concrete syntax. Dans Model Driven Architecture - Foundations and Ap- plications, First European Conference (ECMDA-FA 2005), pages 190–204.

T. Green and M. Petre. (1996). Usability analysis of visual programming environments: a cognitive dimensions framework. Journal of Visual Languages and Visual Computing, 7, pages 131-174.

T. Green, A. Blandford, L. Church, C.R. Roast, et S. Clarke (2006). Cognitive dimensions: Achievements, new directions, and open questions. Journal of Visual Languages & Computing , pages 328-365.

J.C. Grundy, J . G. Hosking, J. Huh, K. Na-Liu Li. Marama: an Eclipse Metatoolset for generating multi-view environments. In proceedings of the 30th International Conference on Software Engineering. ACM, 2008.

K. Henricksen, J. Indulska (2004) Modelling and Using Imperfect Context Information, Proc the CoMeRea workshop at Percom'2004, Orlando, USA.

A. Kleppe (2007) A language description is more than a metamodel. Dans Fourth International Workshop on Software Language Engineering, Nashville, USA,.

J. Krogstie (2003) Evaluating UML using a generic quality framework. Dans UML and the unified process, pages 1–22. IGI Publishing, Hershey, PA, USA.

P. Laforcade, Thierry Nodenot et Christian Sallaberry (2005). "Résultats et perspectives d'un travail exploratoire mené en modélisation et méta-modélisation UML pour la conception de situations d'apprentissage." Numéro spécial "Conceptions et usages des plates-formes de formation" de la revue STICEF (Sciences et Technologies de l'Information et de la Communication pour l'Education et la Formation), volume 12

X. Le Pallec, S. Dupuy-Chessa (2011) Intégration de métriques de qualité des modèles et des métamodèles dans l'outil ModX, Inforsid 2011, papier court

X. Le Pallec, S. Dupuy-Chessa (2012) Intégration de métriques de qualité des diagrammes et des langages dans l'outil ModX, In Conférence en IngénieriE du Logiciel (CIEL), Renne.

N. Mandran, S. Dupuy-Chessa, A. Front, D. Rieu, Démarche centrée utilisateur pour une ingénierie de langages de modélisation de qualité , revue des Sciences et Technologies de l'Information, série Ingénierie des Systèmes d'Information, numéro spécial Evaluation des Systèmes d'Information, volume 18, numéro 3, p. 65 -94, août 2013

S. L. Star. (1989) The structure of ill-structured solutions: boundary objects and heterogeneous distributed problem solving. Dans Distributed Artificial Intelligence (Vol. 2), Morgan Kaufmann Publishers Inc. San Francisco, CA, US, pages 37 – 54

S. L. Star (2010) Ceci n'est pas un objet-frontière! Réflexions sur l'origine du concept. Dans Revue d'anthropologie des connaissances, Vol 4, n° 1, pages 18-35

O. Lindland, G. Sindre and A. Solvberg (1994). Understanding Quality in Conceptual Modeling. IEEE Softw. 11, 2, March 1994, 42-49.

D. L. Moody (2009) The "physics" of notations : Toward a scientific basis for constructing visual notations in software engineering. IEEE Trans. Software Eng., 35(6):756–779.

T. Nhan Luong, S. Laborie, T. Nodenot (2011). A Framework with Tools for Designing Web-based Geographic Applications. Dans the Eleventh ACM Symposium on Document Engineering (DocEng 2011). Googleplex, Mountain View, USA Pages 33-42

The Nhan Luong, Patrick Etcheverry, Christophe Marquesuzaà, Thierry Nodenot (2012). A visual programming language for designing interactions embedded in web-based geographic applications Dans the 17th ACM International Conference on Intelligent User Interfaces (IUI 2012). Lisbon, Portugal, Pages 207-216

T. Nodenot, P. Laforcade and X. Le Pallec (2007). Visual Design of coherent Technology-Enhanced Learning Systems: a few lessons learnt from CPM language. Handbook of Visual Languages in Instructional Design; Theories and Practices. L. Botturi and T. Stubbs, Hershey, PA: IDEA Group: 254-280.

T. Nodenot (2008). Scénarisation pédagogique et modèles conceptuels d'un EIAH : Que peuvent apporter les langages visuels ?, Revue Internationale des Technologies en Pédagogie Universitaire (RITPU) / International Journal of Technologies in Higher Education (IJTHE). Special Issue "Scénariser l'apprentissage, une activité de modélisation" 7(4): 85-103.

P. Mohagheghi et J. Aagedal (2007) Evaluating quality in model-driven engineering. Dans Proceedings of the International Workshop on Modeling in Software Engineering, MISE '07, pages 6–, Washington, DC, USA, IEEE Computer Society.

OCL2, OMG object constraint language, v2.2 (2010)

S. Patig (2008) A practical guide to testing the understandability of notations. Dans Proceedings of the fifth Asia-Pacific conference on Conceptual Modelling - Volume 79, APCCM '08, pages 49–58, Darlinghurst, Australia, Australian Computer Society, Inc.

S. Pinker (1990) A Theory of Graph Comprehension,"Artificial Intelligence and the Future of Testing, R. Freedle, ed., Lawrence Erlbaum and Assoc., pp. 73-126.

I. Ráth, A. Ökrös, D. Varró (2010) Synchronization of Abstract and Concrete Syntax in Domain-Specific Modeling Languages. Software and System Modeling 9:453-471

M. Rossi et S. Brinkkemper (1996) Complexity metrics for systems development methods and techniques. Information Systems, 21(2) :p. 209–227.

B. Selic (2008) UML 2 specification issue 6462, http://www.omg.org/issues/issue6462.txt, 2003, updates dating until 2008.

K. Siau et Y. Tian (2001) The complexity of unified modeling language: A goms analysis. Dans Proceedings of the International Conference on Information Systems (ICIS 2001), pages 443–448.

P. Trompette et D. Vinck. (2009) Retour sur la notion d'objet-frontière. Dans Revue d'anthropologie des connaissances, Vol. 3, n° 1, pp. 5-27

L.Wouters, M.-P. Gervais (2013) Notation-Driven vs Metamodel-Driven Development of Domain-Specific Modeling Languages: an Empirical Study. Symposium on Applied Computing, SAC 2013. ACM.

Index des auteurs