Dauphine | PSL
UNIVERSITÉ PARIS

INFORSID

37ÈME ÉDITION

INFORSID 2019

INFormatique des ORganisations et
Systèmes d'Information et de Décision

11 ▸ 14
Juin 2019
Université
Paris-Dauphine

Président du Comité de Programme
Philippe ROOSE
LIUPPA, Université de Pau et des Pays de l'Adour

Présidente du Comité d'Organisation
Elsa NEGRE
LAMSADE, Université Paris-Dauphine

CONSTRUIRE LES SI POUR LA TRANSFORMATION DES ORGANISATIONS À L'ÈRE DE L'INNOVATION NUMÉRIQUE

http://inforsid.fr/Paris2019/

îledeFrance
cnrs
MIDO
MATHEMATIQUES ET INFORMATIQUE
DE LA DECISION ET DES ORGANISATIONS

Table des matières

Préface

Les Systèmes d'Information (SI) sont au cœur de toute entreprise ou organisation. Si comme M. Jourdain, on fait parfois des SI sans le savoir, on se rend vite compte du besoin de formalisation au fur et à mesure de l'accroissement d'activités des entreprises/organisations et/ou de leurs propres évolutions.

C'est avec grand plaisir que j'écris cette préface pour l'édition des actes de la trente-septième édition du congrès INFORSID (INFormatique des ORganisations et Systèmes d'Information et de Décision). Chère à mon cœur, la pluridisciplinarité peut s'exprimer ici car les SI sont à la croisée de nombreux domaines de l'informatique, et pas uniquement en informatique par ailleurs.

L'organisation d'un congrès annuel est un temps fort de la communauté des SI où les échanges et présentations favorisent les enrichissements mutuels et la mise en perspective de nos recherches. C'est aussi un lieu de rencontres et d'animation de la communauté informatique en France où de ces agitations naissent de nombreuses idées que l'on retrouve par la suite dans des projets, thèses ou autres productions.

En 2019 la conférence INFORSID a accueilli trois conférenciers qui nous ont fait l'honneur d'accepter notre invitation.
Il s'agit de Françoise Berthoud (CNRS) qui nous présente « Numérique : menace ou espoir pour l'environnement ? », une réflexion sur le numérique et vient gratouiller notre (non)-éco-responsabilité ; Frédérique Laforest (LIRIS) « La mécanique des flux » avec l'objectif de présenter les incontournables flux de données situés au cœur de domaines comme les smart grid, smart building, industrie 4.0, e-agriculture et Gilles Dawidowicz (Google) qui nous parle de « Cloud et d'IA au service de la géo-intelligence » afin de faire entrapercevoir la puissance de son apport dans les domaines des objets connectés et de l'information géographique.

La conférence s'anime également avec trois ateliers autour de l' « évolution des SI : vers des SI Pervasifs ? », du « Génie logiciel et qualité de l'internet des objets (QualityIot) » et des « Systèmes d'information et de décision, et démocratie des organisations » ainsi que deux journées thématiques « MaDICS-ADOC - Données de l'Histoire de l'art et d'Archéologie » et « MADICS – LEMON - anaLysE et dynaMique des messages et cONversations radicales sur Internet ». Ces cinq événements viennent enrichir la conférence en mettant le focus sur des problématiques prégnantes ou émergentes.

Ce congrès a décerné également un prix de thèse, en réalité, cette année deux prix car sur 14 soumissions le jury n'a pu départager deux excellentes propositions. Félicitations aux lauréates Jessie Carbonnel pour son « analyse formelle de concepts : un cadre structurel pour l'étude de la variabilité de familles de logiciels » et Elena V. Epure avec la thèse intitulée « Automatically Modeling Conversations as Processes of Interrelated Speech Intentions ». Félicitons au passage les encadrants, car une thèse est à la fois un travail individuel et collectif (oui ça va ensemble !).

Cette année, le congrès INFORSID a reçu 45 soumissions d'articles dont 7 issues d'articles déjà acceptés dans des conférences internationales. Les auteurs sont issus de différents pays (France, Tunisie, Algérie, Belgique, Allemagne, Arménie, Gabon, Espagne, Autriche, Colombie, Qatar, Suisse, Etats Unis). Dix-huit articles ont été acceptés. Comme chaque année, les articles couvrent un large panel des problématiques autour des systèmes d'information, des exigences aux mises en œuvre, des données aux processus métier, du génie logiciel à l'intelligence artificielle.
Parmi ces articles, deux ont fait l'objet d'un accompagnement par un sherpa pour aboutir à la version présente dans ces actes et deux papiers ont été acceptés en tant que papiers courts.

Le processus de sélection des articles s'est déroulé en plusieurs phases. Dans un premier temps, chacun des articles soumis a été évalué par trois à quatre membres du Comité de Programme. Puis, les membres du Conseil du Comité de Programme ont réalisé une méta-évaluation sur chaque article dont les avis divergeaient, coordonné des discussions entre les membres du Comité de Programme.

Avant de clôturer cette préface, je remercie les membres du comité de programme international (7 nationalités). Ils donnent de leur temps, apportent leur crédibilité et réalisent des retours constructifs. C'est une brique essentielle pour une bonne conférence.

Je tiens à chaleureusement remercier les membres du bureau de l'association INFORSID, sous la présidence de Franck Ravat, pour m'avoir confié l'organisation scientifique du congrès et pour leur assistance et implication tout au long de cette année. C'est un honneur. Merci à Agnès, Cécile, Régine, Christian (Oncle Picsou – © Elsa).

Je tiens également à remercier :
- les auteurs pour l'énergie mise dans la rédaction des articles ; les conférenciers venus nous présenter les articles sélectionnés ;
- les membres du conseil du comité de programme, « les sages », pour leurs remarques et conseils à toutes les étapes de la mise en place du congrès ;
- les conférenciers invités pour avoir accepté de nous faire partager leur savoir et leurs expériences ;
- les porteurs des ateliers et de journées thématiques pour leur investissement et leur dynamisme à organiser ces rencontres enrichissantes et originales et qui participent à la fois au dynamisme de la conférence mais également à son renouvellement ;
- les participants au congrès pour faire vivre et promouvoir notre communauté.

Une conférence c'est aussi des personnes qui réalisent un travail de fourmi, qui, s'il est réussi, ne se voit pas, ce qui est d'autant plus frustrant. Je souhaite ainsi terminer par un immense remerciement à Elsa Negre, ainsi qu'à Brice Mayag et Sonia Toubaline de l'Université Paris-Dauphine pour l'organisation sans faille et … et …leur bonne humeur.

Très bonne conférence à tous,

Philippe ROOSE

Président du comité de programme

INFORSID 2019

37ème édition
11 au 14 juin 2019

Comités

Le comité de la 37e édition d'INFORSID est composé par les responsables de l'organisation ainsi que les membres du comité de programme et les membres du conseil du comité de programme.

Comité de programme

Président : Philippe Roose, Université de Pau et des Pays de l'Adour

Membres :
Ikram Amous, Université de Sfax, Tunisie
Pierre-Emmanuel Arduin PSL, Université Paris-Dauphine, Laboratoire DRM (UMR CNRS 7088)
Hakim Bendjenna, Université de Tebessa, Algérie
Isabelle Borne, Université Bretagne Sud, Vannes
Guillaume Cabanac, IRIT - Université Paul Sabatier Toulouse 3
Sylvie, Calabretto, LIRIS - INSA Lyon
Judith Cardinale, Universidad Simón Bolívar, Venezuela
Corine Cauvet, University of Aix-Marseille
Max Chevalier, IRIT, Université de Toulouse
Raja Chiky, ISEP, Paris
Eric Dubois, Luxembourg Institute of Science and Technology
Sophie Dupuy-Chessa, LIG, Grenoble
Cyril Faucher, L3I, La Rochelle
Cécile Favre, Université de Lyon, ERIC Lyon 2
Jérôme Gensel, LIG, Université Grenoble-Alpes
Marianne Huchard, LIRMM, Université de Montpellier
Sergio Ilarri, Université de Zaragoza, Zaragoza, Espagne
Arantza Illarramendi, Université du Pays Basque, San Sébastien, Espagne
Frédérique Laforest, Laboratoire Hubert Curien, Université Jean Monnet, Saint Etienne
Régine Laleau, LACL, Paris-Est Créteil
Olivier Le Goaer, LIUPPA, Université de Pau et des Pays de l'Adour
Kathia Marçal de Oliveira, LAMIH, Valencienne
André Miralles (Irstea - UMR Tetis)
Ludovic Moncla, LIRIS, Lyon
Yannick Naudet, Luxembourg Institute of Science and Technology, Luxembourg
Oscar Pastor, Universitat Politècnica de València, Espagne
Jolita Ralyté, University of Geneva, CUI (Centre Universitaire d'Infomatique)
Claudia Roncancio, LIG, Grenoble
Rajaa Saidi, INSEA, Maroc
André Sales Fonteles, Indiana Wesleyan University, USA
Dalila Tamzalit, Université de Nantes
Virginie Thion, ENSAT, IRISA, Lannion
Hervé Verjus, USBM, Annecy
Robert Viseur, Faculté Polytechnique de l'Université de Mons, Belgique
Karine Zeitouni, David, Université de Versailles

Conseil du Comité de programme

Isabelle Wattiau - ESSEC/CNAM, Paris
Régine Laleau, LACL, Paris-Est Créteil
Olivier Teste, IRIT, Toulouse
Christine Verdier, LIG, Grenoble
Christian Sallaberry, LIUPPA, Pau
Thierry Delot, Crystal, Lille
Khalid Benali, Loria, Nancy
Carine Souveyet, CRI, Paris
Dalila Tamzalit, Nantes
Mireille Blay-Fornarino, I3S, Nice

Comité d'organisation

Présidente : Elsa Negre, Université Paris-Dauphine

Membres :
Brice Mayag, Université Paris-Dauphine
Sonia Toubaline, Université Paris-Dauphine

Conférences Invitées

Numérique : menace ou espoir pour l'environnement ?

Francoise Berthoud

Résumé

Dans notre société du tout numérique et à l'heure où la transition écologique se fait encore attendre en dépit de son urgence, la question se pose du rôle du numérique. Comment le numérique aggrave ou améliore les impacts comme la pollution, le réchauffement climatique, l'effondrement de la biodiversité ou l'épuisement des ressources. Chacun a son idée ; il s'agira ici de poser quelques éléments scientifiques pour mieux comprendre et identifier les leviers pertinents pour changer, dans le bon sens !

La mécanique des flux

Frédérique Laforest

Résumé

Les flux de données sont de plus en plus présents dans les applications phares de notre époque (smart grid, smart building, industrie 4.0, e-agriculture, etc.). Leur prise en compte efficace requiert de repenser leurs modèles de représentation, leurs langages de manipulation, ou encore les moteurs de gestion et traitement de ces données. Je présenterai une synthèse des efforts de recherche menés depuis plus de dix ans dans ces domaines, et ouvrirai des perspectives vers les points qui restent encore à traiter.

Le Cloud et l'IA au service de la géo-intelligence.

Gilles Dawidowicz

Résumé

90% des données numériques existantes ont été générées ces deux dernières années. Et la production continue de s'accélérer. La révolution technologique en cours apportée par le Cloud Computing et l'intelligence artificielle n'épargnera aucun secteur d'activité. Nous ferons un point sur les fondamentaux du Cloud et de l'IA puis illustrerons le propos d'exemples à grand renfort d'objets connectés et d'information géographique.

Aide à la décision basée sur le contexte pour former des groupes d'apprenants pertinents

Elsa Negre, Marie-Hélène Abel

1. *Paris-Dauphine University, PSL Research Universities, CNRS UMR 7243, LAMSADE, 75016 Paris, France*
   elsa.negre@dauphine.fr
2. *Sorbonne Universités, Université de Technologie de Compiègne, CNRS UMR 7253 Heudiasyc, Compiègne, France*
   marie-helene.abel@hds.utc.fr

RÉSUMÉ. *Travailler en groupe vise à dépasser le résultat qui pourrait être obtenu par la simple somme de résultats obtenus individuellement. À cet effet, la définition du groupe est un élément clé : comment choisir les membres de ce dernier, sur quels critères les identifier ? Dans notre travail, nous nous concentrons sur le processus de formation de groupes d'apprenants en tenant compte de leurs caractéristiques et plus généralement du contexte dans lequel ils évoluent. Nous précisons ce que nous entendons par contexte avant de présenter notre approche d'aide à la décision basée sur le contexte pour former des groupes d'apprenants pertinents. À l'aide d'un exemple, nous montrons comment utiliser notre processus de classification multicritère pour l'aide à la décision.*

ABSTRACT. *Working in groups aims to exceed the result that could be obtained by the simple sum of results achieved individually. To this end, the definition of the group is a key element: How to choose the members of the latter, on what criteria to identify them? In our work we focus on the process of forming groups of learners taking into account the characteristics of learners and more generally the context in which they evolve. We specify what we mean by context before presenting our context-based decision support to form relevant groups of learners. Using a motivating example, we illustrate how to use the multi-criteria classification process for decision support.*

MOTS-CLÉS : *Contexte, Aide à la décision (multicritère), Ensembles approximatifs (Dominance-based Rough Set Approach), Environnement collaboratif de travail, Apprentissage en ligne*

KEYWORDS: *Context-aware computing, Decision support, Multi-Criteria Decision Analysis, Dominance-based Rough Set Approach, Collaborative Work Environment, e-learning*

Le travail en groupe a été facilité avec l'arrivée des technologies web 2.0 dont l'usage, dans ce but, tend à se généraliser. Leur utilisation permet la récolte de données diverses et variées qui peuvent être exploitées à différentes fins comme l'aide à la décision. Si l'intérêt de travailler en groupe n'est plus à prouver pour la réalisation de certaines tâches, l'atteinte des résultats visés et/ou la réussite de la collaboration dépend de la composition du groupe. Le processus de l'élaboration de cette dernière reste complexe. Un groupe est généralement formé en fonction de l'impact des caractéristiques de chaque membre sur la cohésion du groupe ou sur la compatibilité des membres du groupe, puis en fonction de la manière dont le groupe interagit. Ces caractéristiques sont ce que nous appelons les informations/données contextuelles de chaque membre du groupe. Les travaux présentés ici peuvent relever des Environnements Informatiques pour l'Apprentissage Humain (EIAH), qui sont conçus dans le but de favoriser l'apprentissage humain, c'est-à-dire la construction de connaissances chez un apprenant. De notre point de vue, un EIAH peut être vu comme un système d'information qui est un ensemble organisé de ressources : individus, infrastructures matérielles, procédures, plateforme numérique, données...

La composition du groupe affecte de nombreux aspects de la réalisation des objectifs pour lesquels il a été créé, tels que l'efficacité avec laquelle les membres du groupe travaillent ensemble et la quantité de connaissances pertinentes qu'ils peuvent partager. Ainsi, dans le domaine de la formation, l'apprentissage par projet, par exemple, nécessite de travailler en groupes et donc de constituer ses groupes. Lorsqu'il faut composer des groupes pour un projet, certaines tâches décisionnelles sont nécessaires, certaines dépendent du contexte de l'apprenant, telles que: (a) définir les objectifs d'apprentissage du projet, (b) décider de la configuration du groupe, (c) sélectionner les membres du groupe, (d) élaborer un plan d'urgence au cas où l'appartenance à un groupe change au cours du projet; et d'autres consistent à obtenir des informations contextuelles telles que : (e) identifier les caractéristiques pertinentes des membres du groupe en fonction des objectifs d'apprentissage du projet (Wang *et al*., 2007).

Dans le cadre de notre travail, nous souhaitons assister les enseignants pour les aider à composer leurs groupes d'apprenants dans le cadre d'un apprentissage par projet, en tenant compte de contraintes et d'informations contextuelles. Les apprenants utilisent la même plateforme informatique comme support à la collaboration grâce à laquelle il est possible de tracer les activités qui y sont effectuées. Sachant que l'usage de la plateforme est récurrent, nous avons des informations/données relatives aux collaborations passées ainsi que sur les utilisateurs de la plateforme (niveaux de connaissance, ...). Nous sommes donc en mesure, à travers la plateforme, d'offrir à l'enseignant un ensemble d'informations sur les apprenants qu'il peut utiliser pour définir ses critères de sélection qui auront un impact sur la composition des groupes [1]. Nous distinguons les critères propres au groupe (nombre de membres, mode de fonctionnement, ...) des critères sur ses membres (âge, niveau de formation, ...). Notre processus d'aide à la décision propose à l'enseignant un ensemble de groupes d'apprenants (i.e.

1. Nous nous concentrons ici sur les critères de l'enseignant. Dans nos travaux futurs, ceux des apprenants pourront être pris également en compte, donnant ainsi une dimension "multi-décideur", à notre approche.

un ensemble d'alternatives pour regrouper les apprenants) qui satisfait les critères et les contraintes (cet ensemble peut être vide). L'enseignant peut alors :

– Valider une alternative : l'enseignant pourra accéder aux informations contextuelles ayant conduit à la sélection de chaque apprenant dans un groupe donné,

– Refuser toutes les alternatives : il sera demandé à l'enseignant de lever/renforcer des critères et/ou des contraintes pour relancer la recherche de groupes.

Dans le domaine du travail collaboratif facilité par les technologies du web 2.0 et du Big Data, dans (Negre, Abel, 2019), nous proposons donc notre processus pour former des groupes d'apprenants pertinents sur la base d'informations contextuelles, c'est-à-dire un processus d'aide à la décision multicritères (basé sur les ensembles approximatifs, DRSA - *Dominance-based Rough Set Approach* (Greco *et al.*, 1999)). L'un des points forts de notre proposition est la possibilité de définir des critères basés sur les activités réalisées lors de l'utilisation d'une plateforme numérique et enregistrées sous la forme de traces d'interaction. A titre d'exemple, si nous considérons un enseignant désirant répartir 6 apprenants en groupes de deux (contrainte forte). Mathématiquement, il existe 15 combinaisons de 2 éléments parmi 6 (C_6^2). Si l'enseignant a pour critères : (i) les membres d'un groupe doivent être géographiquement proches et (ii) leur score de synchronisme [2] doit être proche de 0 alors, en fonction des données contextuelles connues sur les 6 apprenants $a_i, \forall i \in [1, 6]$ (voir ci-dessous), notre approche proposera une seule alternative (parmi les 15) qui satisfera au mieux la contrainte et les deux critères, à savoir $\{(a_1, a_3); (a_2, a_6); (a_4, a_5)\}$.

Apprenant	Taux de synchronisme	Ville
a_1	0,5	Paris
a_2	-1	New York
a_3	0	Paris

Apprenant	Taux de synchronisme	Ville
a_4	1	New York
a_5	0	New York
a_6	-1	Paris

Dans nos travaux futurs, nous pensons appliquer notre processus en exploitant les traces d'interaction sémantique. Nous prévoyons de le tester dans le cadre de la plateforme MEMORAe (Abel, 2015) et de définir des critères tels que les membres ayant déjà travaillé ensembles sur un sujet donné.

Bibliographie

Abel M. (2015). Knowledge map-based web platform to facilitate organizational learning return of experiences. *Computers in Human Behavior*, vol. 51, p. 960 - 966.

Greco S., Matarazzo B., Slowinski R. (1999). The use of rough sets and fuzzy sets in mcdm. In T. Gal, T. J. Stewart, T. Hanne (Eds.), *Multicriteria decision making: Advances in mcdm models, algorithms, theory, and applications*, p. 397–455. Boston, MA, Springer US.

Negre E., Abel M. (2019). Context-based decision support to form relevant groups of learners. In *23th IEEE int. conf. on computer supported cooperative work in design, CSCWD.*

Wang D., Lin S. S. J., Sun C. (2007). DIANA: A computer-supported heterogeneous grouping system for teachers to conduct successful small learning groups. *Computers in Human Behavior*, vol. 23, n° 4, p. 1997–2010.

2. Le score de synchronisme est obtenu par l'analyse des traces d'interaction des apprenants et correspond au rapport entre les nombres de traces synchrones et asynchrones et le nombre total de traces d'interaction.

RFreeStem: A multilanguage rule-free stemmer

Xavier Baril [1], Oihana Coustié [2], Josiane Mothe [3], Olivier Teste [4]

1. *Airbus Opération SAS,*
316 route de Bayonne, Boite postale interne M3014, 31060 Toulouse cedex, France
xavier.baril@airbus.com

2. *Airbus Opération SAS,*
316 route de Bayonne, Boite postale interne M3014, 31060 Toulouse cedex, France
oihana.coustie@irit.fr

3. *IRIT*
118, Route de Narbonne, 31062 Toulouse cedex 04, France
josiane.mothe@irit.fr

4. *IRIT*
118, Route de Narbonne, 31062 Toulouse cedex 04, France
olivier.teste@irit.fr

ABSTRACT. *With the large expansion of available textual data, text mining has become of special interest. Due to their unstructured nature, such data require important preprocessing steps. Among them, stemming is a popularly used preprocessing method that extracts the root of the words. However, the most popular algorithms are based on the application of rules, and therefore highly language-related. We propose a new approach, the RFreeStem, that is rather based on corpus and can therefore be applied on many languages.*

RÉSUMÉ. *La grande disponibilité de données textuelles a rendu populaire les recherches de fouille de texte. Parmi les importants prétraitements nécessaires, la racinisation s'est imposé comme une étape incontournable. Pourtant, les algorithmes les plus utilisés sont basés sur l'application successives de règles. Cette construction les rend fortement dépendants de la langue d'application. Nous proposons ici une nouvelle approche, le RFreeStem, qui se base sur le corpus étudié et promet de pouvoir être appliqué à plusieurs langues.*

KEYWORDS: *Text mining, information retrieval, Sentiment analysis, Stemmer, NLP*

MOTS-CLÉS : *Fouille de texte, recherche d'information, Analyse de sentiments, racinisation*

---

1. Introduction

In the past decades, the overwhelming progress in technologies and Web researches has lead to a high increase of available machine readable documents (Moral *et al*., 2014). Such unstructured data represent a huge potential of information to retrieve. Yet, the absence of structure, along with their high dimensionality make them difficult to process, with an important amount of necessary pre-treatment. Among the numerous pre-processing tasks for text analysis, the idea of stemming the words in the textual data has emerged since the late sixties (Lovins, 1968) and is now almost systematically used (Marcos-Pablos, García-Peñalvo, 2019).

Stemming can be described as the process of replacing a word by its morphological root — the *stem*. Such an algorithm is referred to as a *stemmer*. The stem can be a sub-string or a concatenation of sub-strings of the word, or even a modified sub-string, as in (Porter, 1980), who stems *happy* to *happi*. More than the obtain stem itself, the main result of a stemmer stands in the groups created by regrouping words identically stemmed. Such groups are often referred to as *conflation* groups (Frakes, Baeza-Yates, 1992). For example, (Porter, 1980) would conflate *happy* and *happier*, both stemmed to *happi*. Having words stemmed together automatically reduces the size of the vocabulary since it removes the morphological variants of the words of a lexicon (Jivani *et al*., 2011).

The assumption made here is that morphologically related words have similar meanings and represent the same idea. However, this hypothesis is questionable, that is why many research papers decided to work on lemmatizing algorithms instead. (Moral *et al*., 2014) describes *lemmatization* as linguistic-based approach that relies on Part of Speech(PoS) and context. They are less dependants on the morphological variants of the words than stemmers are. To do so, they often draws on dictionaries or thesauri. Even if they offer prospectively more accurate results, their high complexity make them unpopular. Lemmatization will therefore not be detailed furthermore in this paper, since it is another field of research. We will rather focus on stemmers, reputed to be faster and more popular (Jivani *et al*., 2011).

Indeed, stemming has become an almost inescapable step in some text mining tasks. Among them, *Information Retrieval* (IR) algorithms automatically analyze documents to extract information to answer user queries (Hull, 1996): since the stem represents a broader concept than the word, more documents are expected to match the query. Other text mining fields extensively use stemming pre-processing. (Jivani *et al*., 2011) describes it as a requirement for many *Natural Language Processing* (NLP) applications, including text clustering and categorization or sentiment analysis — the process to automatically describe the sentiments of the author of a message.

However, the most widely used stemmer are based on rules to be applied on the words, in order to remove irrelevant parts. These rules rely on the morphological forms of the words and thus are necessarily language-specific. Whereas highly-spoken languages have their rules well described, it is difficult to find an implementation of those algorithms for rare languages or dialects. Moreover, the number of rules, as well

as the quality of the stemmer, seem to highly depend on the language it is applied on (Kraaij, Pohlmann, 1996). To be more specific, linguistics distinguish two different types of language structures (Moral *et al*., 2014):

– Analytic (or isolating) languages, with little morphology structures, where the variants are more likely to be expressed by helper words than by word variations (e.g. Chinese, Vietnamese, modern English).

– Synthetic languages, with strong morphology structures. Among them we find: (i) The inflective language, like Latin languages (French, Italian) or Germanic languages (German, Dutch, Swedish, (Braschler, Ripplinger, 2004)). They have numerous prefixes and suffixes applied to modify the grammatical nature or even the meaning of the words.(ii) The agglutinative language, like Finnish, Japanese or Basque. In those languages the suffixes do not only represent inflections but can also be the concatenation of other morphemes. For example, in basque, *exte* means *house* and *etxean* means *in the house*.

Regarding those different categories of languages, (Jivani *et al*., 2011) affirms that stemming has proven more efficient on languages with complex morphology, that is to say on synthetic languages. (Braschler, Ripplinger, 2004) come to the same conclusion with its study on Germanic ones. Yet, on such languages, a great number of rules would be needed, in order to remove all the affixes, which would make the stemmer poorly flexible.

Furthermore, stemming evaluation can be performed by comparing its results to a set of validated labels. In this case, further than the traditional precision and recall, (Paice, 1994) defines the *overstemming* and *understemming* errors. Overstemming error counts the number of words stemmed together whereas they shouldn't have and understemming error census the words with different stems that should have been stemmed together. Paice proposes the calculation of indicators based on such metrics. However, they all assume the access to an objective label set, which is difficult to census in practice, especially on rare languages.

In the light of these observations, our contribution consists in a rule-free stemmer, that can be applied on multiple languages. To cope with the previously mentioned evaluation issues, we will evaluate our stemmer by measuring its impact on the results of data mining tasks. We will also assess what we call *compression power* — the ability of a stemmer to compress the size of a vocabulary. We will measure it thanks to the famous *Index Compression Factor* (Sirsat *et al*., 2013), that asses the strength of stemmers. Finally, we shall be paying special attention to the stem produced. Although (Jivani *et al*., 2011) mentions that stemmers do not need to be words from the language, it can nevertheless be crucial that the stems are humanly readable in tasks like query expansion or dictionary look-up (Hull, 1996).

In accordance with those matters, we first propose to census the different types of stemming methods found in the literature in section 2, before describing ours, the RFreeStem in setion 3. We will finally describe, in section 4 an evaluation framework to assess the results of our stemmer on different text mining tasks and languages.

2. Related work

The literature offers an important variety of stemming algorithms that can mostly be categorized in two types of approach : (a) the rule-based methods, (b) the corpus-based methods. The following subsections propose to census some of the most important studies on stemming algorithms in those two categories.

2.1. Rule-based methods

Most of the well-known stemming methods are based on the application of rules. Among them, *affix stripping* gathers procedures to remove the suffixes and, sometimes, the prefixes of the words. In 1968, (Lovins, 1968) proposed a first suffix stripping method. Aggressive and fast, the algorithm matches the end of the word with the longest element of a suffix list, and applies rules to modify the matching end. Lovins's algorithm is reputed to be unreliable since its list of suffixes is based on a very technical-oriented vocabulary. Yet, technical and scientific words tend to derive from Latin languages and therefore to be highly inflectional, whereas common modern English is rather isolating[1].

The now most popular algorithm was soon proposed by (Porter, 1980). With 5 successive steps, it is a bit slower than its 2-step predecessor, but has experimentally proven more accurate (Willett, 2006), (Paice, 1996). Initially, Porter exclusively defined those successive rules to stem English words. Yet, its popularity motivated the implementation of `snowball`(Porter, 2001), a detailed framework that enables users to implement their own version of Porter stemmer in any language. However, it remains a current subject of study to find such implementations for less common or highly inflectional languages (e.g. Bahasa Indonesian (Tala, 2003), Dutch (Kraaij, Pohlmann, 1994)).

Many other rule-based methods were proposed afterwards. Among them, the light *S stemming* only deals with plural forms, and so presents little compression power. On the other hand, (Paice, 1990) proposed a strong algorithm that applies successive deletion and replacement rules. (Jivani *et al.*, 2011) argues that this method often presents a high over-stemming error. (Dawson, 1974) implemented an adaptation of Lovins method with much more suffixes and rules, which unfortunately makes the algorithm hardly reusable (Jivani *et al.*, 2011). (Krovetz, 1993) proposes an approach based on inflections and derivations, with its Krovetz stemmer, *KSTEM*. It relies on an inflection-free lexicon to withdraw inflections, before analyzing the derivations — variants that change the grammatical nature of words. The author acknowledges that the performance strongly depends on the chosen lexicon. We even add that finding such an exhaustive lexicon might not be feasible for rare language or dialect.

1. English language is actually slightly inflectional, due to its heritage of old English. Thus, modern English is more fit for stemming than purely isolating languages like Mandarin or Indonesian (Moral *et al.*, 2014).

Those rule-based methods are undeniably the most recognized and used in information retrieval, thanks to their fast computation skills and easy implementations (Harman, 1991). Nevertheless, adapting those methods to highly-inflectional languages would lead to a large number of necessary rules. Consequently, recent papers that focus on stemming text from those languages are inclined to look for other approaches than rule-based ones.

2.2. Corpus-based methods

In order to cope with the strong language dependence of rule-based stemming techniques, some corpus-based methods were proposed. Most of them rely on statistical principles. (Hafer, Weiss, 1974) developed a *Successor variety* stemmer which cuts the words in two parts : if the first part belongs to the corpus, this first part becomes the stem. The cutting algorithm is based on the evolution, within the word's letters, of the *successor variety* : the number of distinct characters that follow a string within all the words of the corpus. Thus, the resulting conflations strongly depend on the corpus, and yet, systematically choosing the first part of the word as a stem seems to be a language-dependant approach.

Another method, — the *ngram stemmer*— proposed by (Adamson, Boreham, 1974), uses a common string clustering method to create conflation groups : a hierarchical single linkage clustering of the words. To evaluate the word pairwise distance, the authors used the Dice distance, corresponding to the number of distinct shared digrams — sequence of two consecutive letters. Using an unsupervised clustering method to create conflation groups makes the stemming corpus-dependant but also offers a very flexible way to manage stemming strength. However, single linkage clustering is known for its important algorithm complexity, due to the creation of a quadratic distance matrix. Very similarly, *YASS* (Yet Another Stripping Stemmer), described in (Majumder *et al*., 2007), clusters words with linkage hierarchical methods, but has also implemented word distance metrics that encourage long suffix matching. According to (Jivani *et al*., 2011), they make the method more adapted for languages that are richly suffixed.

As a last interesting example, (Soricut, Och, 2015) proposed an unsupervised corpus-based method that automatically identifies meaningful rules to strip prefixes, with overall good results. The unsupervised learning on the corpus makes this method highly flexible and potentially adaptable to many languages.

From our reading of the literature, some stemming features appear to be interesting and have constituted the baseline of the method we propose. A rule-free method offers the possibility for the stemmer to be applied on many languages. Moreover, an unsupervised hierarchical clustering method does not need any prior knowledge and enables a high flexibility in the stemmer strength. Finally, the use of ngrams seems particularly fit for strings classification. The following section describes how those features were implemented for the RFreeStem algorithm.

3. The RFreeStemer: a new stemmer

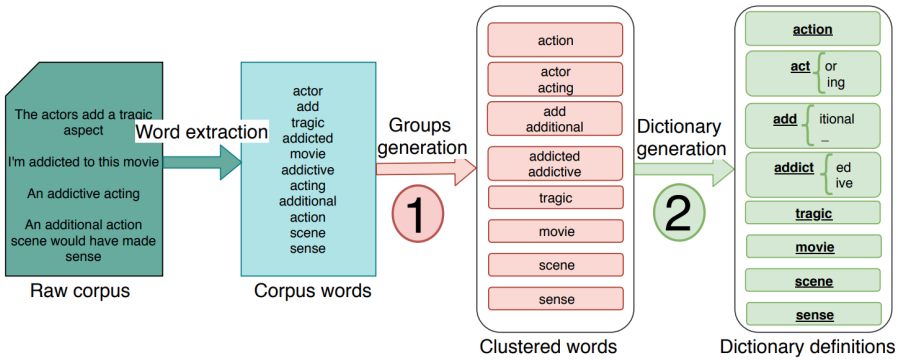


Figure 1. The general approach of RFreeStem stemmer. First step (1) is presented in section 3.1. Second step (2) is detailed in section 3.2

To answer the detected issues of existing stemmers, we propose a new approach with RFreeStem stemmer. Our corpus-based algorithm clusters the words of a corpus in order to create conflation groups. It is therefore an unsupervised learning method, that does not need any external knowledge. The proposed clustering is hierarchical — which allows flexibility in term of stemmer strength — and based on ngram similarity. Moreover, the clustering method promises to be much faster than any distance-matrix-based clustering. Indeed, instead of creating such a quadratic matrix, the algorithm only runs through the ngrams once. The RFreeStem stemmer is divided in two steps: after extracting all the words from the corpus, the group generation step clusters the words into conflation groups, and the dictionary generation step creates a readable dictionary out of the stemmed groups. Those steps are resumed in Figure 1 and describes in the following subsections.

3.1. Word clustering

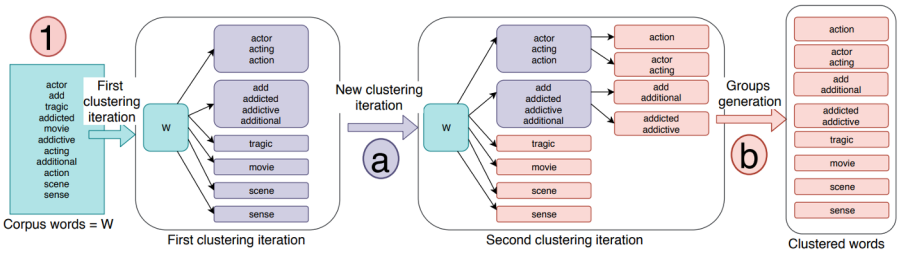


Figure 2. The clustering method (1) with (a) the recursive division (see 3.1.1) and (b) the dendrogram cutting (see 3.1.3

The first step of our process consists in clustering the corpus words. After the extraction of all the words of the corpus, they are processed to be regrouped by con-

flation. To do so, we use a hierarchical clustering method — generally more adapted for text clustering— with a divisive approach : starting from the whole set of words, we recursively split it into clusters, as illustrated in Figure 2. We call those recursive steps *divisive steps*, and describe them in the subsection 3.1.1. We therefore discuss the choice of the number of divisive step needed in section 3.1.3. Indeed, deciding how to cut the dendrogram directly impacts the conflation groups and the stemmer strength.

3.1.1. Divisive step algorithm

At each divisive step, a set of several words is divided into several groups. The algorithm 1 formally describes this process.

Algorithm 1: Divisive step algorithm

```
   input  : A set of (unique) words W, an integer n;          // n as in ngram
   output: A set of word groups
1  ngram.set ← findnGram(words = W, n = n);
2  ngram.set ← scoreFunction(set = ngram.set);
3  ngram ← first(ngram.set);
4  G ← emptyGroup();
5  while ngram != last(ngram) and size(W) > 0 do
6  |   matching.words = grep(pattern = ngram, set = W);
7  |   if 0 < size(matching.words) < size(W) then
8  |   |   remove(elementsToRemove = matching.words, set = W);
9  |   |   g ← createGroup(elements = matching.words);
10 |   |   addGroup(element = g, group = G);
11 |   end
12 |   ngram = next(ngram);
13 end
14 return G;
```

The algorithm uses the functions : (i) `findnGram` that extracts all the unique ngrams within a set of words (ii) `scoreFunction` that sorts a set of ngram and returns it. The implementation of that function is described in subsection 3.1.2. (iii) `first`, `last` and `next` which respectively returns the first, last or next element of a set (iv) `createGroup` that creates a group with words (v) `emptyGroup` and `addGroup` which respectively initializes a empty set of clusters and a cluster to a set (vi) `grep` which returns the words that match a pattern.

In the first divisive step, the input set of words is the whole corpus, whereas in each intermediate ones, the input is an already formed cluster. For an input W, $w \in W$ is a word of the set, on an alphabet $\mathcal{A}$. If $w = a_1 a_2 .. a_k$, where k is the length of w and $\forall i, a_i \in \mathcal{A}$, then a ngram of w is a subset $a_i .. a_{i+n}$, $i \in [\![1; k-n]\!]$. Notice that w exactly has $k - n + 1$ ngrams. We propose to extract all the distinct ngrams of all the words. Let $\mathcal{N}$ be such a set of ngrams. Our method then sorts the ngrams (subsection 3.1.2), which are then browsed exactly once. For each $ng \in \mathcal{N}$, all the words that contain ng are clustered together. This one-browse clustering offers an interesting decrease in computational execution time, compared with traditional distance-based clustering algorithms. In addition, the condition $\texttt{size}(W) = 0$ — all the words are clustered — is generally reached before the end of the browse, stopping the execution.

3.1.2. Scoring function choice

It appears from the previously described algorithm that the definition of the function that sorts the ngram is crucial — We call it the *scoring function*. Indeed, it is designed to give a score to each ngram, knowing they will be browsed according to these scores. A ngram with a high score will be responsible for the clustering of all the words that contain this ngram together. We define the *generated group* of an ngram as the set of words that contain this ngram. Regarding our algorithm, the generated groups of well-scored ngrams will become clusters, and the final generated groups will gather words that will be stemmed together. Hence, if the ngram *tion* has a better score than *clas*, then *classification* will be clustered with the *tion*-words — like *imitation*, *attention* — instead of being united with words like *class* or *classify*. To avoid this situation, we discuss interesting features that can define a "good" ngram:

– The number of words containing this ngram, which also is the size of the generated group. A small value only groups a few words together, which conflicts with the divisive approach of our hierarchical clustering. On the other hand, the biggest-size ngrams — the most frequent ones — are often common affixes (e.g. *tion*). A middle ground size could be the mean size of all the generated groups.

– The homogeneity of the generated group, that can be evaluated with: (i) A string distance within the generated group (e.g. edit distance) (ii) The number of ngrams in the generated group. An homogeneous group is expected to have few ngrams. (iii) A ngram distance on those ngrams, like Dice distance (vi) The number of common letters within the generated group words. This idea relates to the further dictionary creation we propose : a high number of common letters means a longer stem.

– The heterogeneity of the generated group. One measure could be the number of other generated groups that contain the ngram. Indeed, we easily imagine that *tion* is likely to appear in almost every natural conflation group. However, the relation "ng_1 belongs to the generated group of ng_2" is reciprocal. Therefore, this quantity is precisely the number of ngrams in the generated group of ng_1 (see homogeneity).

– Another intuition while refining this algorithm on English datasets was to penalize ngrams that were likely to be affixes — ngrams whose positions in the words are often either at the beginning or at the end. We feel however that this criteria is highly language-related and thus, in conflict with our rule-free principle.

We finally propose to score the ngrams with the following function :

$$s(ng_i) = \frac{1}{2} \times (\texttt{mean}(\texttt{DiceDistance}(\mathcal{N}_i)) + ||\mathcal{N}_i| - id.size|)$$

where $\mathcal{N}_i \subset \mathcal{N}$ is the set of ngrams in ng_i's generated group, $id.size$ is the ideal size of a group, namely the mean size of all the generated groups and `DiceDistance` function returns all the pairwise Dice distance for a set of ngrams. The first member represents the group homogeneity : a low Dice distance between the ngrams of a generated group reflects a group with similar words. The second member penalizes the gap between the ideal and the real size of a generated group. Based on several

experiments, we are confident that this scoring function will grant a good score to the ngrams reflecting the real stem of words.

3.1.3. Cutting the dendrogram into clusters

Once the dendrogram is generated, a valuable question is to know how to deduce the groups. Indeed the example of figure 2 shows three possible variants : the example chooses to stop after the second iteration. Though, we could have stopped at the first one, or continue to split another time. We define the *depth* of the clustering as the chosen number of iterations. A high depth value leads to a very light stemming, with many words stemmed alone or in small groups. This configuration will decrease the risk of overstemming. On the contrary, a low value of depth leads to large and fewer groups. Such an aggressive stemming decreases the understemming error rate, and increases the compression power of the stemmer. Since we think that depth value is highly data-dependent, we do not fix it, and will rather compare the experimental results obtained for different values of depth.

For this first version of the RFreeStem stemmer, we only consider homogeneous value of depth : the clustering process stops at the same iteration for each group. Though, it could have been more accurate to allow groups to have different depths. This approach would have required the definition and implementation of a stopping criteria, to automatically decide when a cluster is considered homogeneous enough. We let that interesting question for a future work extension.

3.2. Dictionary generation

The second step of our algorithm aims to generate a dictionary out of the groups previously created. In each homogeneous group $\mathcal{W}_g$, this step consists in finding the fix and variable parts in the words of the conflation group. A fix part is defined as a sequence of consecutive letters that can be found in each word of the group. For instance, the group formed by the two words *classification* and *unclarified* has two fix parts : `cla` and `ifi`. The variable parts are simply the non-fix parts of the words. In the example, we would respectively find {`ss`, `cation`} and {`un`, `r`, `ed`}. The goal is to obtain a dictionary censusing the stems of the group (fix parts) and all the possible values taken by the variable parts, as shown in the last part of Figure 1. In the minimal example of the words *classification* and *unclarified*, we actually want to align the matching variable parts, to obtain :

$$\left\{ \begin{array}{l} \text{un} \\ \_ \end{array} \right. \text{cla} \left\{ \begin{array}{l} \text{r} \\ \text{ss} \end{array} \right. \text{ifi} \left\{ \begin{array}{l} \text{ed} \\ \text{cation} \end{array} \right.$$

Yet, the number of variable is not constant. In order to be able to align them, we refine our definition : a variable part v_{i_j} is an ngram between the i^{th} and $i + 1^{th}$ fix parts, with $i \in [\![0; F]\!]$, where F is the number of fix parts in a group, and $j \in [\![1; |\mathcal{W}_g|]\!]$. For instance here, we have $v_{1_1} =$ `r` and $v_{1_2} =$ `ss`. For a given i, the vector $v_i = \{v_{i_j}, j \in [\![1; |\mathcal{W}_g|]\!]\}$ is the set of all the value taken by the variable parts.

To illustrate the difficulty of finding the fix parts, we extend our example to $\mathcal{W}_g =$ { *classification*, *declaration*, *unclarified*, *clarity*, *claimed*, *cleansing*, *classic* }. Of course, obtaining such a heterogeneous cluster is not desirable. Though, in case the scoring function did not behave as expected, the dictionary generation method has to handle such heterogeneity. To find fix parts, we first look for letters that are in every words of $\mathcal{W}_g$. We call *message types* such fix letters — `c`, `l`, `a` and `i` in the example. For the word *clarity*, we can immediately deduce the variable parts. Though, for the words *declaration*, *classification* and *unclarified* multiple occurrences of the message types `c`, `i` and `a` appear and we need to know which one is a real fix part. The following table census the frequencies of the fix parts in each words.

	clarity	*claimed*	*cleansing*	*classical*	*unclarified*	*declaration*	*classification*	min.
`c`	1	1	1	1	1	1	2	1
`l`	1	1	1	1	1	1	1	1
`a`	1	1	1	1	1	2	2	1
`i`	1	1	1	1	2	1	3	1

We define the *perfect examples* as the words with the minimum frequencies of each message type — the first four words of the table. We aim to chose, for the words *unclarified*, *declaration* and *classification*, the occurrences of the message types that are fix parts. We call α_i the i^{th} occurrence of letter α in a word : in *unclarified*, $\texttt{i}_1$ is the `i` between the `r` and the `f`. To decide which occurrences of the message types to keep, we first look at the order in which the fix parts appear. Based on our perfect examples, the order is always $\texttt{c}, \texttt{l}, \texttt{a}, \texttt{i}$. This helps us chosing the c_1 occurrence of *classification* rather than c_2. However, it does not promote any of the `a` and `i` occurrences in the three problematic words. We therefore add another feature to the selection : the gap between the fix parts for each perfect example. This value cannot be calculated for the other words, since we would not know which occurrence to chose.

gap lengths	*clarity*	*claimed*	*cleansing*	*classical*	mean	interquartile range
`c - l`	1	1	1	1	1	0
`l - a`	1	1	2	1	1.25	0.25
`a - i`	2	1	3	2	2	0.5

The mean of the gaps can be seen as the ideal gaps that should separate the fix parts. For *unclarified*, the gap between `a` and $\texttt{i}_1$ is 2, and between `a` and $\texttt{i}_2$ is 4. Therefore, we successfully select the $\texttt{i}_1$ occurrence. Moreover, for *classification* we have the following gaps between occurrences of `a` and `i` :

`a`-occurrence	`i`-occurrence	occurences gap	expected gap	difference with expected
$\texttt{a}_1$	$\texttt{i}_1$	3	2	1
$\texttt{a}_1$	$\texttt{i}_2$	5	2	3
$\texttt{a}_1$	$\texttt{i}_3$	9	2	7
$\texttt{a}_2$	$\texttt{i}_3$	2	2	0

However, the huge gap between a_2 and l will fortunately have us chose a_1 — since we take the mean of the gaps. Nevertheless, it not sufficient for the word *declaration* :

a-occurrence	observed		difference with expected		mean difference
	l - a	a - i	l - a	a - i	
a_1	1	4	0	2	1
a_2	3	2	2	0	1

Indeed, the two occurrences of a are assigned the same score. We therefore add a last feature to our decision method : a measure of gap stability. In the previous example, we observe that the gap between the letters l and a has more stable values than the gap between a and i. Hence, the difference between the real and the ideal values of the gap should take into account the reliability of the gap. Thus, we propose to calculate the interquartile range of each gap (last column of table 2). This interquartile is inversely proportional to the coefficient attributed to the gaps. In our example, since the relation l − a is quite stable, it will be highly penalized for a_2 to have an extra gap of 2. The a_2 occurrence will finally be chosen as a fix part. With all the defined process of fix part selection, we simply deduce the variable parts as the variable letters and the unselected common letters. We obtain, in our example, the wanted result :

$$\left\{\begin{array}{l} \text{un} \\ \text{de} \\ \_ \end{array}\right. \text{cl} \left\{\begin{array}{l} \text{e} \\ \_ \end{array}\right. \text{a} \left\{\begin{array}{l} \text{r} \\ \text{ns} \\ \text{ss} \\ \text{arat} \\ \_ \end{array}\right. \text{i} \left\{\begin{array}{l} \text{ty} \\ \text{med} \\ \text{ng} \\ \text{cal} \\ \text{fied} \\ \text{on} \\ \text{fication} \end{array}\right.$$

The stemmed extracted from this group could then be _cl_a_i_. In practice, we hope not to obtain such an heterogeneous group. In a well-formed group we can hope to have more intelligible stemmers like _classif_ or _clar_. This new version of the generated stem could enhance string comparison results, which might be needed in NLP tasks. Moreover, dictionary look-ups will be easier for the user.

4. Evaluation

We propose to assess the results of our implementation of the RFreeStem stemmer. We aim to demonstrate that our stemming method offers better accuracy results than the traditionally used Porter stemmer, while proposing a better compression power. To do so, we propose to run our stemmer and Porter's on raw data before the application of a NLP task.

4.1. Evaluation framework

Our evaluation is divided in two discussions : the efficiency of our stemmer in two different text mining tasks in English, and the comparison of this efficiency depending

on the language of the corpus. For each discussion, we will compare the different versions of our algorithm — due to the possible values of the depth of the cut — with both Porter's and the unstemmed data.

Task 1: Text categorization is a popular text mining task that aims at automatically determining the class of a document within a corpus. To test the efficiency of our stemmer as a preprocessing step for text categorization, we implemented our method on the AustLII case report corpus [2]. We retrieved 6480 citations out of 717 different XML files. Those files contain both the text of the legal citation and a manually labelled category, among the following :

cited	referred to	applied	followed	considered	discussed	Others(12)
2847	1372	715	529	361	323	170

Our dataset is highly imbalanced since 43.9% of the citations have the label *cited*, whereas half of the classes only contains 1.25% of the data. Imbalanced datasets are reputed to be harder to process, so it will be interesting to see if our stemmer can help in that matter.

Task 2: Sentiment analysis has become a very popular task, in particular with the high expansion of social networks and their textual data. The aim is to evaluate sentiments in human written comments or reviews. To evaluate the ability of RFreeStem to can improve the results of this complex task, we chose the commonly used movie review dataset [3]. Firstly mentioned in (Pang, Lee, 2004), the corpus is still popularly used today (Ba *et al.*, 2016), (Shen *et al.*, 2018). It contains 10662 perfectly balanced reviews of movies, with 18196 different terms.

Since the state-of-the-art seems to agree that stemming is more adapted to inflectional languages, we propose to compare our stemming on related datasets from different languages. We therefore studied the amazon reviews in French and German[4] for sentiment analysis task (**Task 2**). Among the large amount of available data, we chose, for each language, a sample of 2000 reviews, respectfully of the proportion of positive and negative labels. Since the products are evaluated with a rating from 1 to 5, we extracted the binary labels with the following simple transformation rule : if the rating is strictly over 3, assign to positive class. Otherwise, assign to negative. We obtain the following datasets:

Language	#reviews	Positive values		Negative values		#features
French	2000	1634	81.7%	366	18.3%	11 583
German	2000	1704	85.2%	296	14.8%	16 143

2. http://www.austlii.edu.au/ (Galgani *et al.*, 2012) uses it for categorization
3. http://www.cs.cornell.edu/people/pabo/movie-review-data/
4. https://s3.amazonaws.com/amazon-reviews-pds/readme.html

Similarly to the full datasets, the reviews are highly imbalanced, which allows us to test the robustness of our stemmer. We also note that German has more than 30% features more than French. Indeed, besides being highly inflective, German language tends to compose new words by juxtaposing two smaller ones. Moreover, it uses declensions to construct its articles and adjectives, which adds a large variety of word variants. German therefore seems like a perfect fit for stemming algorithms.

Finally, since our method is corpus-based, we claim that it is not language-dependant and could be applied on any language, especially the rarely documented ones, where it is hard to find a Porter-like stemmer. We propose to study a rare Roman Urdu data set[5] for a 3-class sentiment analysis (**Task 2**):

#reviews	Positive		Neutral		Negative		#features
20229	6013	29.8%	8929	44.1%	5287	26.1%	31888

The process we apply is as follow : for each dataset, we apply the different versions of our method to the whole raw data and therefore generate a stemming dictionary where each entry is a {word: stem} couple. As mentioned before, we want to observe the effect of the depth of the cut on the results. Therefore, we generate as many dictionary as we have candidate depths for the cut. Thanks to the study of (Harman, 1991), we have the idea of chosing to use 4-grams, that seem to be the most relevant, both on analytic and inflective languages.

We then run a supervised classification algorithm on different versions of the data : the raw data (pre-processed without stemming), the Porter data (pre-processed and stemmed with Porter stemmer) and all the d-RFreeStem data (pre-processed and stemmed with our algorithm, cut with a depth of d). We choose to train a Naive Bayesian classifier on 70% of the data, respecting the proportion of the labels. The 30% left are predicted with the trained classifier and the results are compared with their labels. To select the classifier input features, we simply calculate the document-term matrix, and sort the features in decreasing frequency order. For each configuration, we run several versions of the feature selection step. While i is less than $max(feature.frequencies)$, we select the features whose frequency is at least i. Each value of i gives a different number of features as input of the classifier. For each stemmer, we run several times all the features selection step to have a robust mean value.

4.2. Results and discussions

By running 20 times all the version of our algorithms, Porter's and the raw version, for all the possible feature selection configurations, we obtain the results presented in that subsection. As a measure of result improvement, we calculate the traditional F-measure, as well as the accuracy. We also evaluate the ICF value, to measure the

5. http://archive.ics.uci.edu/ml/datasets/Roman+Urdu+Data+Set, (Sharf, Rahman, 2018)

compression power of the understudied stemmer. Firstly, let us compare the accuracy

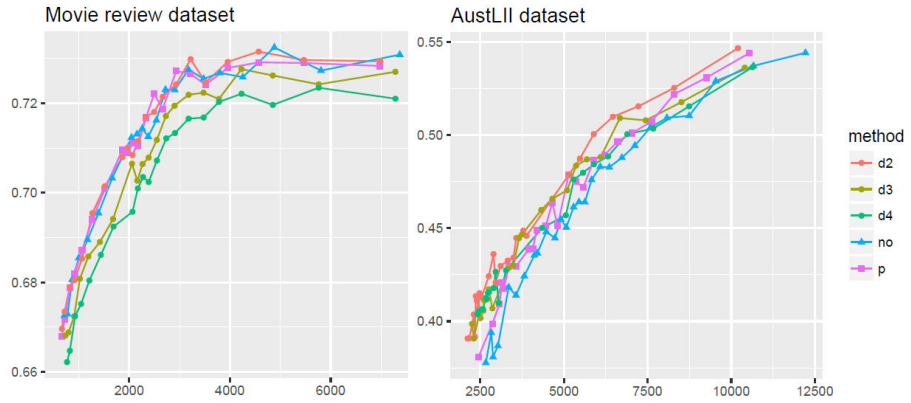


*Figure 3. Accuracy comparison for different #features between (**Task 2**) movie reviews binary classifier and (**Task 1**) AustLII multinomial classifier*

obtained in the two English mining tasks. We observe slighlty different results on those two evaluations. In the classification task, stemming systematically improves the results. The most aggressive versions of our algorithm 2-RFreeStem performs better than Porter, while 3-RFreeStem and Porter present similar results. Yet, the accuracy range is rather law : multiclass categorization seems to be a hard task to be applied on documents, and stemming can significantly help preserving a correct level of prediction. On the other hand, the sentiment analysis results are lukewarm. Almost all the stemming methods deteriorate the accuracy compared to the unstemmed version, apart from our most aggressive stemmer 2-RFreeStem. The movie review is also reputed to be a difficult task and it seems that stems might have masked the discrimination out of the variant forms of the words. Moreover, as explained before, better results are expected on inflective languages. To prove it, we compare this last result to the Amazon review ones.

Since French is a highly inflective language, each root has many variants and thus many words can be stemmed together while respecting their meanings. However, let us note that this corpus regroups human informal speeches. In informal speech, French tends to be less inflective : we would rather say *not happy* than *unhappy*. Nevertheless, our two most aggressive versions perform well and better than Porter (Figure 4). Similar results are observed with German texts, where 4 of our algorithms have at least one configuration where they outperform Porter, which struggles to reach the unstemmed performances. Actually, there is a clear segregation between Porter or the unstemmed version and our methods. This makes us think that our stemming algorithm is well adapted for German. Indeed, Porter stemming does not include any treatment for complex composed words, whereas (Braschler, Ripplinger, 2004) argues that decompounding task is an even more decisive factor than stemming for the German language. On the other side, our algorithm is theoretically capable of retrieving such complex linguistic structures, which can explains those encouraging results. Eventu-

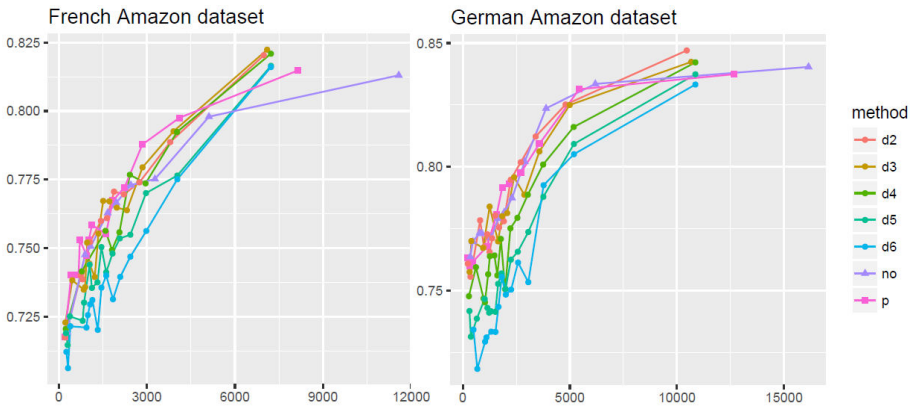


Figure 4. Accuracy comparison for different #features between French and German Amazon reviews dataset for a binary sentiment analysis

ally, those two highly inflective languages clearly show the compression power of our stemmer. Indeed, the maximum number of features for the unstemmed version is far greater than for our methods, due to the important vocabulary reduction performed. We naturally also detect a slight difference in that matter between the d-versions of our algorithms : the lower d value, the lower maximum number of features.

Finally, we propose to see how our method behaves on a poorly documented language as Urdu. For this language, we did not have a Porter version implemented and therefore only compared our results to the unstemmed one in Figure 5. We discover that only our most aggressive version can compete with the raw data. The improvement observed is far from being significant, but at least, we take satisfaction in proposing a method that will not deteriorate the accuracy for this left-behind language.

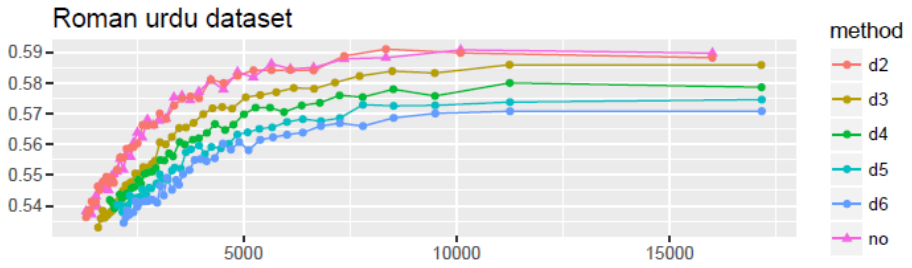


Figure 5. Comparison of the accuracy of the binary classifier prediction on Roman Urdu data for different #features

To conclude, the following table summarizes the evaluation features we wanted to observe: (i) F1 is the best F1 measure (over #features), (ii) is similarly the best accuracy value and ICF is the index compression factor of the method. For the English

version, we took the AustLII data since it is wider and shows more sensitivity to stemming. The following results are presented in percentage.

	English			French			German			Urdu		
	F1	Acc	ICF	F1	Acc	ICF	F1	Acc	ICF	F1	Acc	ICF
No	**33**.4	54.4	0	70.3	81.3	0	70.3	84.0	0	**59.1**	60.0	0
P	**33.4**	54.4	44.6	69.9	81.5	27.0	71.1	83.7	14.4	_	_	_
d2	33.0	**54.6**	**145**	**70.7**	82.0	**109**	71.2	**84.7**	**131**	**59.1**	**60.1**	**114**
d3	32.5	53.6	58.9	70.4	**82.2**	46.3	**71.5**	84.2	53.4	58.6	59.5	34.4

The best result are often obtained by our 2-RFreeStem version. This fact might be disappointing : in this version, only one clustering step is achieved, and no hierarchical tree is ever created. With this actually flat clustering, our method only consists in clustering the words depending on one of their 4-grams. A very similar method is proposed by (Pande *et al.*, 2013), yet they sort the ngrams very differently. However, our 3-RFreeStem version has sometimes proven more accurate (accuracy in French, F1-measure in German), and we argue that an even better solution could be found in refining our cutting algorithm. According to the results, for every groups created in the first splitting iteration, we could either chose to stop there or to split one more time, depending on an homogeneity criteria that is still to be defined.

5. Conclusion

Stemming is still systematically used as preprocessing step for text mining tasks. Since the sensitivity of stemming is highly language-dependant, we have proposed a multilanguage rule-free stemmer, based on corpus data. We have shown that it outperforms the famous Porter algorithm, and that its performance are even more spectacular on inflectional languages. An improvement of our method could be done by refining our dendrogram cutting method, thanks to a group homogeneity criteria. Moreover, instead of always working with 4-grams, we will soon try to enable the variation of the n length. We could also have proposed a wider evaluation framework by (i) analyzing the RFreeStem performance on agglutinative or rare languages (ii) assessing our stemmer accuracy improvement on Informational Retrieval, and even, since it is multilingual, on Cross Language Information Retrieval (iii) confronting our stemmer to other corpus-based methods, like (Soricut, Och, 2015). We let those interesting questions opened for a future contribution.

References

Adamson G. W., Boreham J. (1974). The use of an association measure based on character structure to identify semantically related pairs of words and document titles. *Information storage and retrieval*, Vol. 10, No. 7-8, pp. 253–260.

Ba J. L., Kiros J. R., Hinton G. E. (2016). Layer normalization. *arXiv preprint arXiv:1607.06450*.

Braschler M., Ripplinger B. (2004). How effective is stemming and decompounding for german text retrieval? *Information Retrieval*, Vol. 7, No. 3-4, pp. 291–316.

Dawson J. (1974). Suffix removal and word conflation. *ALLC bulletin*, Vol. 2, No. 3, pp. 33–46.

Frakes W. B., Baeza-Yates R. (1992). *Information retrieval: Data structures & algorithms* (Vol. 331). Prentice Hall Englewood Cliffs, NJ.

Galgani F., Compton P., Hoffmann A. (2012). Knowledge acquisition for categorization of legal case reports. In *Pacific rim knowledge acquisition workshop*, pp. 118–132.

Hafer M. A., Weiss S. F. (1974). Word segmentation by letter successor varieties. *Information storage and retrieval*, Vol. 10, No. 11-12, pp. 371–385.

Harman D. (1991). How effective is suffixing? *Journal of the american society for information science*, Vol. 42, No. 1, pp. 7–15.

Hull D. A. (1996). Stemming algorithms: A case study for detailed evaluation. *Journal of the American Society for Information Science*, Vol. 47, No. 1, pp. 70–84.

Jivani A. G. *et al*. (2011). A comparative study of stemming algorithms. *Int. J. Comp. Tech. Appl*, Vol. 2, No. 6, pp. 1930–1938.

Kraaij W., Pohlmann R. (1994). Porter's stemming algorithm for dutch. *Informatiewetenschap*, pp. 167–180.

Kraaij W., Pohlmann R. (1996). Viewing stemming as recall enhancement. In *Sigir*, Vol. 96, pp. 40–48.

Krovetz R. (1993). Viewing morphology as an inference process. In *Proceedings of the 16th annual international acm sigir conference on research and development in information retrieval*, pp. 191–202.

Lovins J. B. (1968). Development of a stemming algorithm. *Mech. Translat. & Comp. Linguistics*, Vol. 11, No. 1-2, pp. 22–31.

Majumder P., Mitra M., Parui S. K., Kole G., Mitra P., Datta K. (2007). Yass: Yet another suffix stripper. *ACM transactions on information systems (TOIS)*, Vol. 25, No. 4, pp. 18.

Marcos-Pablos S., García-Peñalvo F. J. (2019). Information retrieval methodology for aiding scientific database search. *Soft Computing*, pp. 1–10.

Moral C., Antonio A. de, Imbert R., Ramírez J. (2014). A survey of stemming algorithms in information retrieval. *Information Research: An International Electronic Journal*, Vol. 19, No. 1, pp. n1.

Paice C. D. (1990, November). Another stemmer. *SIGIR Forum*, Vol. 24, No. 3, pp. 56–61. Retrieved from http://doi.acm.org/10.1145/101306.101310

Paice C. D. (1994). An evaluation method for stemming algorithms. In *Proceedings of the 17th annual international acm sigir conference on research and development in information retrieval*, pp. 42–50.

Paice C. D. (1996). Method for evaluation of stemming algorithms based on error counting. *Journal of the American Society for Information Science*, Vol. 47, No. 8, pp. 632–649.

Pande B. P., Tamta P., Dhami H. S. (2013). Generation, implementation and appraisal of an n-gram based stemming algorithm. *arXiv preprint arXiv:1312.4824*.

Pang B., Lee L. (2004). A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd annual meeting on association for computational linguistics*, p. 271.

Porter M. F. (1980). An algorithm for suffix stripping. *Program*, Vol. 14, No. 3, pp. 130–137.

Porter M. F. (2001). *Snowball: A language for stemming algorithms.* Retrieved from http://snowball.tartarus.org/

Sharf Z., Rahman S. U. (2018). Performing natural language processing on roman urdu datasets. *INTERNATIONAL JOURNAL OF COMPUTER SCIENCE AND NETWORK SECURITY*, Vol. 18, No. 1, pp. 141–148.

Shen T., Zhou T., Long G., Jiang J., Pan S., Zhang C. (2018). Disan: Directional self-attention network for rnn/cnn-free language understanding. In *Thirty-second aaai conference on artificial intelligence.*

Sirsat S. R., Chavan V., Mahalle H. S. (2013). Strength and accuracy analysis of affix removal stemming algorithms. *International Journal of Computer Science and Information Technologies*, Vol. 4, No. 2, pp. 265–269.

Soricut R., Och F. (2015). Unsupervised morphology induction using word embeddings. In *Proceedings of the 2015 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pp. 1627–1637.

Tala F. Z. (2003). A study of stemming effects on information retrieval in bahasa indonesia. *Institute for Logic, Language and Computation, Universiteit van Amsterdam, The Netherlands*.

Willett P. (2006). The porter stemming algorithm: then and now. *Program*, Vol. 40, No. 3, pp. 219–223.

Evaluation de la gouvernance de l'information

Une approche multifacette et multicritères

Jacky Akoka[1], Isabelle Comyn-Wattiau[2]

1. CEDRIC-CNAM et Institut Mines-Telecom BS
2 Rue Conté, 75003 PARIS, France, jacky.akoka@lecnam.net
2. ESSEC Business School
3 Avenue Bernard Hirsch, 95021 CERGY, France, wattiau@essec.edu

RESUME. *Face aux enjeux représentés par l'information et son rôle dans la prise de décision, les entreprises sont de plus en plus amenées à passer du management de cette information à sa gouvernance. L'objectif de cet article est de présenter une approche d'évaluation de cette gouvernance dont l'avantage est de prendre en compte les points de vue des différentes parties prenantes. Cette démarche est fondée sur une évaluation multifacette et multicritères, reposant sur une hiérarchie générique de critères d'évaluation, ensuite spécialisée par partie prenante (management, gouvernance de l'information, métiers). La hiérarchie est organisée selon les dimensions d'un système. La démarche est illustrée par un cas pratique.*

ABSTRACT. *Faced with the challenges represented by information and its role in decision-making, companies are more and more likely to move from the management of this information to its governance. This article presents an approach to evaluate this governance. Its main interest is to take into account the points of view of different stakeholders. This approach is based on a multifaceted and multi-criteria evaluation, using a generic hierarchy of evaluation criteria. The hierarchy is specialized for different stakeholders (management, information governance, business groups). It is organized according to the main dimensions of systems theory. The approach is illustrated on a practical case.*

MOTS-CLES : *gouvernance de l'information, approche multifacette, évaluation multicritères, approche système.*

KEYWORDS: *information governance, multi-facet approach, multi-criteria evaluation, systemic approach.*

1. Introduction

La gouvernance de l'information comporte plusieurs dimensions relatives à son contenu et aux processus qu'elle met en œuvre. Gartner (2019) la définit comme un ensemble de droits, de processus, de normes et de politiques régissant l'information tout au long de son cycle de vie. Ce dernier s'étend de la création de l'information à sa destruction en passant par son stockage, son archivage et bien sûr son utilisation. Elle inclut les processus, les rôles et les politiques, les normes et les mesures qui garantissent l'utilisation efficace des informations afin de permettre à une organisation d'atteindre ses objectifs. L'information est considérée comme une

ressource de l'entreprise. Sa gouvernance nécessite une structure et des processus pour gérer son cycle de vie. C'est donc l'ensemble des structures, des politiques, des procédures, des processus et des technologies mis en œuvre pour gérer les informations au niveau de l'entreprise. C'est un cadre de référence constitué de parties prenantes, de principes, de processus et d'outils qui définissent la raison, le moment et la manière de gérer l'information dans une organisation dans le but de maximiser sa valeur, de répondre aux obligations réglementaires et de réduire son « coût de possession » et les risques qu'elle fait courir à l'entreprise.

Etant donné les enjeux financiers, managériaux, stratégiques et sociétaux de la gouvernance de l'information, il est impératif de procéder à son évaluation de manière récurrente en préservant les points de vue des différentes parties prenantes, notamment (i) les fonctions juridiques et/ou liées au respect de la réglementation, (ii) la Direction des Systèmes d'Information (DSI) chargée notamment des infrastructures et de la gestion des données, (iii) la fonction de gouvernance de l'information qui peut faire partie – ou non – des prérogatives de la DSI, (iv) les entités opérationnelles, y compris les ressources humaines et les finances et (v) le management de l'entreprise.

Ces différentes parties prenantes ont des visions subjectives de la gouvernance de l'information. La prise en compte de ces différents points de vue permet d'obtenir une évaluation plus équilibrée et, par là, d'améliorer la gouvernance du point de vue de chaque partie prenante. Il s'agit donc de préserver plusieurs perspectives dans l'évaluation de cette gouvernance. A cette fin, nous adaptons une approche multifacette (Maynard *et al*., 1999). Cette dernière consiste à considérer la politique de gouvernance comme un système générant des opinions divergentes, notamment sur son efficacité et son efficience. Le résultat d'une évaluation par des groupes multiples ne vise pas à produire une mesure unique, mais permet d'obtenir un résultat final reflétant les visions subjectives. Ainsi, l'évaluation combine une approche multifacette et une hiérarchie multicritères. Dans notre approche, l'évaluation de la gouvernance de l'information est le résultat de l'évaluation des objectifs qu'elle poursuit, des structures qu'elle met en œuvre, des activités qu'elle réalise, de l'interaction avec son environnement et enfin des évolutions qu'elle subit. Certains composants de cette gouvernance sont évalués de manière qualitative, tandis que d'autres le sont de manière quantitative.

Le reste de cet article est organisé comme suit. Nous présentons dans la section 2 un état de l'art sur la gouvernance de l'information et son évaluation. La section suivante est dédiée à la présentation de notre approche. L'application à une entreprise de chimie est décrite dans la section 4. Nous concluons dans la section 5.

2. Evaluation de la gouvernance de l'information : un état de l'art

La recherche sur l'évaluation de la gouvernance de l'information est encore à ses débuts. L'objectif de notre article est de présenter une démarche d'évaluation multifacette et multicritères. A cette fin, nous présentons un état de l'art des principaux concepts de la gouvernance de l'information et de son évaluation.

2.1. Les principaux concepts de la gouvernance de l'information

Il existe de nombreuses définitions de la gouvernance de l'information. Certains auteurs mettent l'accent sur les structures et les politiques (Smallwood, 2014), tandis que d'autres font ressortir les dimensions relatives aux activités ainsi que celles liées aux technologies que les entreprises utilisent (Ballard *et al*., 2014). (Foster *et al*., 2018) mettent l'accent sur le concept de capital informationnel que l'entreprise doit maximiser via sa politique de gouvernance de l'information. La constitution de ce capital informationnel exige des investissements importants qui nécessitent un processus de supervision et de décision permettant de hiérarchiser ces investissements (Bennett, 2017). Nous considérons qu'il s'agit d'un *processus de régulation* dont l'objet principal est de fixer les règles du jeu qui permettront de prendre les meilleures décisions compte tenu du but à atteindre.

Il existe une différence entre la gouvernance des systèmes d'information, des données et de l'information. La *gouvernance des systèmes d'information* (SI) est portée par une finalité : assurer l'alignement des SI sur les objectifs métiers. La *gouvernance des données* comprend les politiques, procédures et processus liés aux données, inclus la modélisation, la cartographie, l'audit, les contrôles et la gestion de la qualité, l'architecture et la gestion des référentiels de données (Bennett, 2017). Elle inclut la définition des données, la gestion des données de référence (Master Data), des métadonnées, le nettoyage et la transformation des données, leur mise en correspondance ainsi que leur classification. Rappelons qu'une donnée devient une information par un processus d'*interprétation* qui fait intervenir les connaissances de l'individu. La donnée dans un *contexte* devient une information. Cette dernière est la base sur laquelle les décisions sont prises et les services fournis. *La gouvernance de l'information*, quant à elle, a pour objectif de garantir que ces informations sont traitées de manière efficace et efficiente à tout moment. En particulier, elle doit aider à assurer la fiabilité des informations. Elle doit protéger la vie privée des parties prenantes de l'entreprise (personnel, clients, etc.) et se conformer aux exigences légales et réglementaires. Elle doit soutenir la prise de décision en veillant à ce que les informations pertinentes puissent être facilement localisées, minimiser les risques d'atteinte à la sécurité des informations et augmenter la rentabilité de l'entreprise en s'assurant que les informations sont éliminées dès qu'elles deviennent obsolètes ou ne sont plus nécessaires. On parle ainsi d'actifs informationnels. Elle a pour objectif de gérer les *informations* de l'organisation dans le but de soutenir la croissance de l'entreprise. De notre point de vue, elle correspond à l'ensemble des politiques et des procédures mises en place au sein d'une entreprise afin d'encadrer la collecte des informations et leur utilisation**.** Elle diffère du *management de l'information* qui constitue la façon dont l'information est gérée tout au long de son cycle de vie (recueil, gestion, stockage, classification, distribution, conservation et destruction). Enfin, la *gouvernance de l'information* est la base de la construction d'un programme de management de l'information juridiquement défendable. Elle constitue la structure et le cadre de la stratégie de management de l'information. Ainsi, on peut affirmer que l'objectif principal de la gouvernance de l'information est celui de *valoriser l'information en en faisant une ressource partagée, fiable et de qualité*, *en vue de la prise de décision.*

2.2. L'évaluation de la gouvernance de l'information

Il n'existe pas, à notre connaissance, d'approche d'évaluation de la gouvernance de l'information. Cette dernière étant un artefact en vue de la décision, nous présentons un état de l'art sur les méthodes d'évaluation en général, puis sur l'évaluation des artefacts et, enfin, sur l'évaluation des systèmes d'aide à la décision.

Une classification des méthodologies d'évaluation a été proposée par Scriven (1967) qui différencie l'évaluation « formative » que l'on applique pendant le processus de développement et d'implémentation d'un artefact, de l'évaluation « sommative » que l'on réalise à la fin de ce processus. Ces deux approches ne donnent aucune indication sur la manière dont l'évaluation peut être effectuée tant en ce qui concerne la stratégie à adopter que les méthodes à utiliser ou encore les critères d'évaluation pertinents. Pour pallier ces limites, (Cronholm et Goldkuhl, 2003) comparent trois démarches d'évaluation. La première, appelée *goal-based*, est fondée sur l'évaluation d'objectifs mesurables faisant généralement appel à des méthodes quantitatives. La deuxième, intitulée *goal-free*, est une méthodologie inductive qui vise à collecter des données sur un grand nombre d'effets réels, puis à évaluer l'importance de ces effets. La troisième, appelée *criteria-based*, effectue l'évaluation selon des listes prédéfinies de critères, lesquels découlent principalement de certaines théories ainsi que de directives ou de normes. Le choix d'une des trois démarches dépend de la motivation qui préside à l'évaluation.

Il existe aussi une littérature, tout aussi abondante, sur l'évaluation des artefacts générés par les sciences de conception (Design Science Research). (Peffers *et al*., 2012) établissent un lien entre les types d'artefacts et les techniques d'évaluation. Une revue de la littérature relative à l'évaluation des artefacts développés dans les démarches de Design Science Research (DSR) est présentée dans (Prat *et al*., 2015). March et Smith (1995) fournissent une liste complète de critères d'évaluation par type d'artefact. (Hevner *et al*., 2004) proposent des critères et une typologie des méthodes d'évaluation. Sonnenberg et vom Brocke (2012) fournissent de nombreux exemples de critères d'évaluation et de méthodes, mais ne relient pas directement les méthodes d'évaluation aux critères. (Cleven *et al*., 2009) caractérisent les approches d'évaluation selon douze dimensions dont aucune ne mentionne les critères. (Peffers *et al*., 2012) analysent les méthodes d'évaluation couramment utilisées par type d'artefact. (Järvinen, 2007) soutient que, dans DSR, l'évaluation met trop l'accent sur le critère d'utilité.

En ce qui concerne l'évaluation des systèmes d'aide à la décision, trois approches ont été proposées. La première, définie par Adelman (1992), comporte trois facettes : technique, empirique et subjective. Toutefois, elle semble souffrir d'un manque d'organisation et ne rentre pas dans le cycle de vie classique des systèmes d'aide à la décision. La deuxième approche, appelée séquentielle, comporte quatre phases : la définition des critères d'évaluation, une évaluation formative suivie par une évaluation des résultats et se conclut par une évaluation sommative (Sojda, 2004). Cette approche est plus complexe et semble être limitée au seul point de vue des évaluateurs. La troisième approche, plus générale, est proposée par (Rhee et Rao, 2008). Elle est fondée sur un modèle global d'évaluation

de l'efficacité d'un système d'aide à la décision indépendamment des technologies ou du domaine d'application. Toutefois, elle souffre d'un manque de précision. Boukhayma et ElManouar (2015) structurent les méthodes d'évaluation en trois catégories : multifacettes, séquentielles et générales.

A notre connaissance, il n'existe pas dans la littérature de démarche d'évaluation de la gouvernance de l'information. On trouve, certes, des cadres de référence permettant d'évaluer la maturité de la gouvernance de l'information dans l'entreprise (Proença *et al*., 2016). L'objectif principal de ces cadres est de mesurer le degré de maturité des entreprises dans la gouvernance de l'information et non d'évaluer cette dernière. L'absence de méthodes d'évaluation de la gouvernance de l'information est sans doute due au caractère récent de cette approche. Notre objectif est précisément de capitaliser sur les méthodes d'évaluation des artefacts des sciences de conception et des approches utilisées pour l'évaluation des systèmes d'aide à la décision. A cet effet, *nous considérons la gouvernance de l'information comme un système, offrant un artefact en vue de la décision*.

3. L'approche multifacette et multicritères

L'approche multifacette permet de prendre en compte une variété de points de vue lors de l'évaluation de la politique de gouvernance de l'information au sein d'une organisation. Elle permet de mettre en œuvre un processus d'évaluation suffisamment souple pour tenir compte de la subjectivité des différentes parties prenantes. L'objectif n'est pas de présenter un point de vue unique. L'application de la démarche nécessite (i) l'identification des parties prenantes, (ii) le développement d'une hiérarchie générique de critères d'évaluation, (iii) l'adaptation de cette hiérarchie à chaque partie prenante, et (iv) l'élaboration d'une procédure d'évaluation.

3.1. Identification des parties prenantes

L'identification de toutes les parties prenantes concernées par l'évaluation est un aspect important de l'approche multifacette. Dans notre cas, nous avons identifié quatre grands groupes de personnes impliquées dans l'évaluation de la gouvernance de l'information : (1) les bénéficiaires du programme de gouvernance, inclus les personnes chargées des fonctions juridiques, celles liées au respect de la réglementation et le personnel des fonctions opérationnelles telles que la finance, la logistique, le marketing, etc. De manière générique, nous les appelons les groupes fonctionnels. (2) la Direction des Systèmes d'Information (DSI) chargée notamment des infrastructures et de la gestion des données, (3) les personnes chargées de la fonction de gouvernance de l'information qui ne sont pas forcément rattachées à la DSI, et (4) le management de l'entreprise, garant de la valeur que peut générer la gouvernance de l'information. Le management doit veiller à inclure la gouvernance de l'information dans les projets de l'entreprise et éviter l'effet de silo dû au manque de collaboration entre les fonctions de l'entreprise concernées par l'usage de l'information. Il doit aussi allouer un budget dédié à la gouvernance de l'information

et considérer cette dernière comme une activité génératrice de valeur et non comme un centre de coûts uniquement. Enfin, il doit – et c'est une condition du succès d'un programme de gouvernance – s'impliquer dans ce dernier.

A noter que l'opinion de ces groupes de personnes est souvent différente. Ces différents groupes représentent des rôles distincts dans le développement et la mise en œuvre de la politique de gouvernance de l'information. Certaines personnes peuvent, bien entendu, jouer plusieurs rôles. Dans tous les cas, il est important de considérer tous les facteurs qui peuvent jouer un rôle lors de l'évaluation et qui reflètent leur point de vue. Nous présentons dans la suite les critères pouvant être utilisés dans l'évaluation de la politique de gouvernance de l'information, préservant les différentes perspectives de l'évaluateur. Nous présentons dans un premier temps l'ensemble des critères d'évaluation indépendamment des parties prenantes. Puis, dans les paragraphes suivants, nous instancions cette hiérarchie générale de critères pour chaque partie prenante.

3.2. Une hiérarchie générique de critères d'évaluation

Plusieurs éléments concourent à considérer la gouvernance de l'information comme un système. Le premier élément est relatif au fait que son objet est la prise de décision, à la fois pour fixer les orientations de l'action dans le domaine de l'information (*les objectifs*), et pour définir et ajuster le cadre de fonctionnement (*structures et environnement*) des activités correspondantes (*les structures et les activités*). De plus, elle est amenée à faire évoluer si nécessaire (*évolution*) les objectifs et les modalités des activités qui lui sont associées. Enfin, la gouvernance de l'information trouve sa signification dans la mise en relation des éléments qui la composent, les objectifs, les structures, l'environnement, les activités et l'évolution qui sont les principales dimensions d'un système (Le Moigne, 2006 ; Skyttner, 2005). A ce titre *la gouvernance de l'information possède toutes les caractéristiques d'un système*.

Le deuxième élément est le fait que la politique de gouvernance de l'information constitue un artefact que l'entreprise développe et met en œuvre pour atteindre ses objectifs. Or, plusieurs auteurs considèrent les artefacts de conception comme des systèmes (Simon, 1996 ; Gregor, 2010). Ils les caractérisent en termes de fonctions (*activités*), *d'objectifs* et d'adaptation (*évolution*). Ils distinguent également la *structure* de l'artefact de *l'environnement* dans lequel il opère. Ainsi, *l'artefact « gouvernance de l'information » possède toutes les dimensions de la forme canonique d'un système*. De ce fait, nous pouvons adopter une vision holistique de son évaluation. Cela nous permet d'organiser les critères d'évaluation en fonction des dimensions fondamentales des systèmes.

Capitalisant sur la théorie des systèmes (Skyttner, 2005), nous reprenons la hiérarchie des critères présentée dans (Prat *et al*., 2015) et nous l'enrichissons grâce à certains des critères proposés dans (Maynard *et al*., 1999). La hiérarchie associe, à chacune des dimensions du système de gouvernance, les critères d'évaluation appropriés (Figure 1).

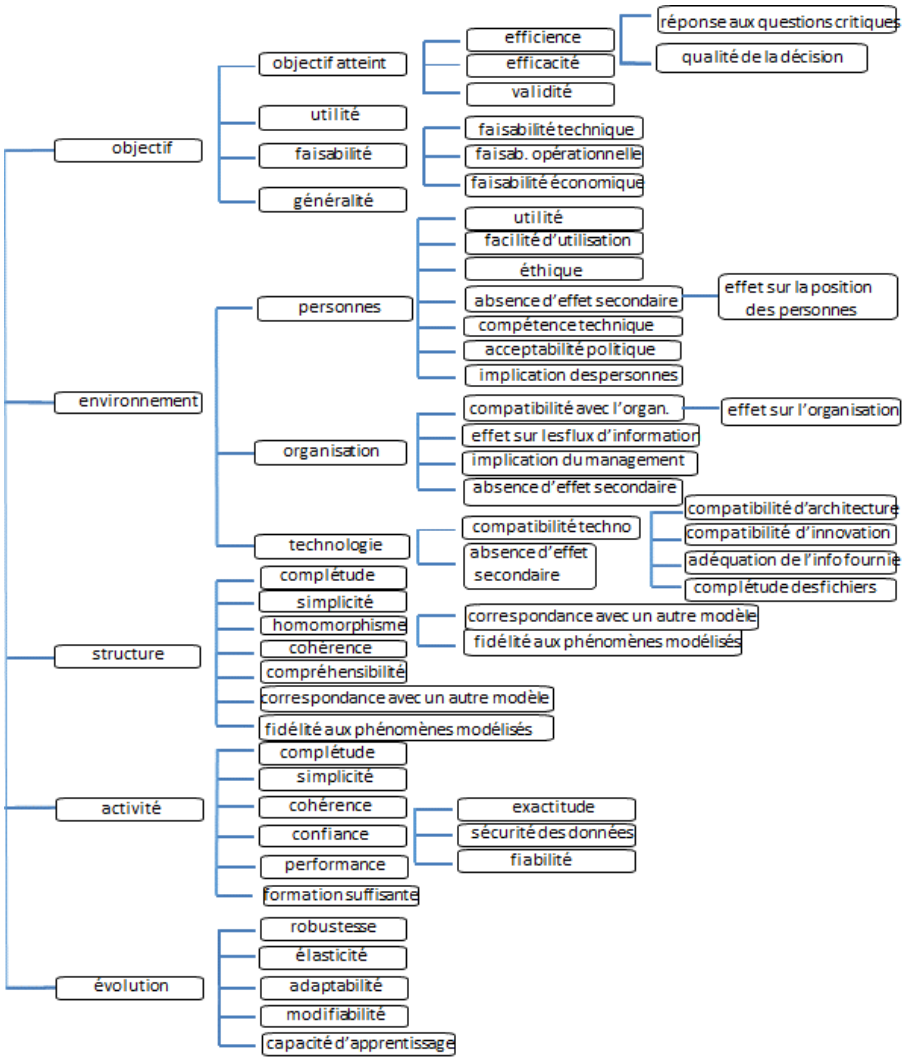


Figure 1. Hiérarchie des critères d'évaluation de la gouvernance de l'information

La première dimension de cette hiérarchie, appelée objectif de l'artefact, est évaluée à l'aide de plusieurs critères : l'efficience mesure si l'artefact a atteint son but (Venable *et al*., 2012), alors que l'efficacité mesure si l'objectif est atteint dans une situation réelle. L'évaluation de l'efficience intègre la capacité de la gouvernance à apporter une réponse aux questions critiques de l'entreprise en matière de management de l'information. Elle comprend aussi une évaluation de l'impact de la gouvernance sur la qualité de la décision. Enfin, la validité indique si l'artefact a atteint *correctement* son objectif (Gregor et Hevner, 2013). L'utilité mesure la différence entre la valeur de la réalisation de cet objectif et le prix payé pour l'atteindre. Il existe trois critères pour la faisabilité : la faisabilité technique

évalue, d'un point de vue technique, la facilité avec laquelle un artefact proposé est construit et exploité. La faisabilité opérationnelle évalue, quant à elle, dans quelle mesure la direction, les employés et les autres parties prenantes soutiennent l'artefact proposé, l'exploitent et l'intègrent. La faisabilité économique détermine si les avantages d'un artefact proposé dépassent les coûts de construction et d'exploitation de cet artefact. Enfin, le critère de généralité fait référence à la portée de l'artefact. Plus la portée de l'objectif est large, plus l'artefact est général (Aier et Fischer, 2011).

La deuxième dimension concerne la cohérence de l'artefact avec son environnement qui comprend les personnes, l'organisation et les technologies (Hevner *et al*., 2004). La cohérence avec l'environnement est aussi appelée cohérence externe (Sonnenberg et vom Brocke, 2012). En ce qui concerne les personnes, l'utilité évalue dans quelle mesure l'artefact a un impact positif sur la performance des tâches réalisées par lui. Elle mesure la qualité de l'artefact utilisé dans la pratique. La facilité d'utilisation (March et Smith, 1995) évalue si l'utilisation de l'artefact par les individus requiert ou non des efforts. L'éthique (Venable *et al*., 2012) évalue le degré de conformité de l'artefact avec les principes éthiques. L'absence d'effets secondaires mesure le degré avec lequel l'artefact est exempt d'impacts indésirables sur les individus à long terme, par exemple l'effet de la gouvernance sur leur position dans l'entreprise. En ce qui concerne l'organisation, le critère de compatibilité avec l'organisation permet de mesurer l'adéquation de l'artefact avec l'organisation et sa stratégie. On évalue aussi l'implication du management, l'effet de la gouvernance sur les flux d'information et l'inexistence d'impact indésirable sur l'organisation à long terme.

La troisième dimension, celle de la structure, est composée des critères suivants : la complétude évalue si la structure de l'artefact contient tous les éléments nécessaires (la politique de gouvernance couvre toute l'entreprise) et les relations entre les éléments. La simplicité évalue si la structure de l'artefact est minimale en termes de nombre d'éléments et de relations entre ceux-ci (March et Smith, 1995). Ainsi, la question est de déterminer si la structure de gouvernance de l'information contient des rôles et des fonctions redondants. On évalue aussi la compréhensibilité de l'artefact. La cohérence, quant à elle, mesure l'harmonie entre les différents éléments de l'artefact. Enfin, l'artefact, considéré comme un modèle, peut être comparé avec un modèle de référence (homomorphisme). L'homomorphisme peut aussi mesurer la fidélité de l'artefact dans la représentation des phénomènes modélisés.

La quatrième dimension, relative à l'activité de l'artefact, est caractérisée par la complétude, la simplicité, la cohérence, l'exactitude, la fiabilité et la performance. La complétude permet de s'assurer que l'artefact contient toutes les activités nécessaires à la réalisation de ses objectifs, autrement dit si la politique de gouvernance couvre toutes les activités afférentes. La simplicité évalue si l'artefact contient le nombre minimal d'activités, sans recouvrement entre elles. La cohérence correspond à l'absence de contradiction entre les activités de l'artefact. L'exactitude indique le degré de concordance entre les résultats de l'artefact et les résultats attendus (Aier et Fischer, 2011). La fiabilité évalue la capacité de l'artefact à

fonctionner correctement dans un environnement donné pendant une période donnée. Enfin, la performance désigne la capacité de l'artefact à remplir ses fonctions dans des contraintes données de temps ou d'espace.

La dernière dimension, celle de l'évolution, se caractérise par la robustesse, l'élasticité, l'adaptabilité, la modifiabilité et la capacité d'apprentissage. La robustesse désigne la capacité de réagir aux fluctuations de l'environnement. L'élasticité correspond à la capacité de l'artefact à gérer des volumes croissants de travail. L'adaptabilité mesure la facilité avec laquelle l'artefact peut fonctionner dans des contextes autres que ceux pour lesquels il a été conçu. La modifiabilité évalue la facilité avec laquelle l'artefact peut être modifié sans introduire de défauts. Enfin, la capacité d'apprentissage est la capacité d'un système à tirer des leçons de son expérience.

Cette hiérarchie de critères est générique et doit être adaptée à chaque partie prenante. En effet, certains critères ne sont pas nécessairement pertinents pour tous les groupes concernés. C'est le principe de base de l'approche multifacette. Nous présentons donc ci-dessous la hiérarchie de critères spécifique à chaque partie prenante.

3.3. La hiérarchie de critères spécifique aux différentes parties prenantes

Pour prendre en considération les visions subjectives de chaque partie prenante, il nous faut adapter la hiérarchie générique successivement pour le management, les groupes fonctionnels et le groupe Gouvernance de l'Information. En ce qui concerne le management, nous développons de manière détaillée les raisons qui nous conduisent à retenir certains nœuds de la hiérarchie générique. Pour les autres parties prenantes, et pour des raisons d'espace, nous nous contenterons d'indiquer les raisons principales.

a. Hiérarchie de critères spécifique au management

En nous fondant sur la théorie des systèmes, nous considérons que tout système est organisé autour d'un ou plusieurs objectifs. Il a une finalité, qui est le but poursuivi par une organisation. Dans notre cas, les préoccupations du management de toute organisation portent principalement sur la manière dont la politique de gouvernance de l'information affecte l'entreprise. Le management doit évaluer le succès ou l'échec de cette politique au regard des objectifs poursuivis par l'entreprise. Il doit déterminer les avantages que la gouvernance de l'information apporte à l'organisation. La qualité des décisions et l'effet de la politique de gouvernance sur l'efficacité organisationnelle sont des éléments qui permettent d'évaluer le point de vue du management sur le succès d'une politique de gouvernance de l'information. La conséquence de cette vision de la finalité de la gouvernance nous amène à conserver dans la hiérarchie les nœuds correspondant à l'*objectif*, à l'exclusion de la *faisabilité technique* qui relève de la mission de la DSI ou de l'équipe en charge de la gouvernance de l'information. En ce qui concerne l'environnement, rappelons que la théorie des systèmes le définit comme l'ensemble

des éléments extérieurs au système considéré, et avec lequel ont lieu des échanges, qu'il s'agisse d'information, de matière ou encore d'énergie. Dans notre hiérarchie, la dimension de l'*environnement* qui concerne en premier lieu le management est celle de l'*organisation*. Nous conservons donc dans notre hiérarchie spécifique les nœuds correspondant à cette dimension et nous excluons les deux autres dimensions : *personnes et technologie.* La dimension *structure* renvoie à l'inventaire des éléments dont se compose le système et à leurs agencements les uns par rapport aux autres. Il nous apparait que cette dimension n'est pas cruciale du point de vue du management. Elle ne sera donc pas retenue dans la hiérarchie spécifique.

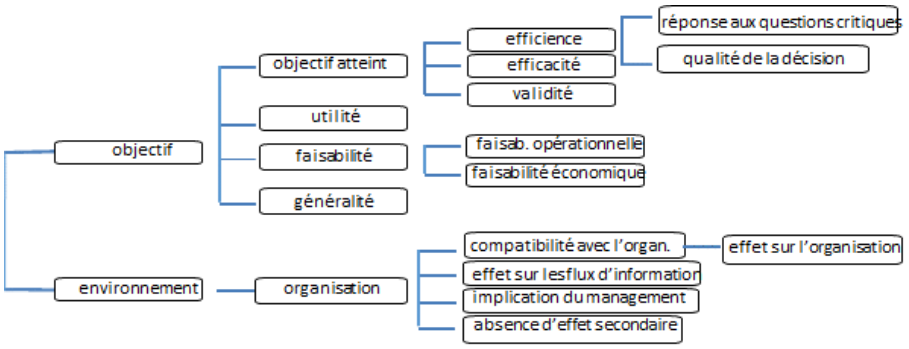


Figure 2. Hiérarchie adaptée au management

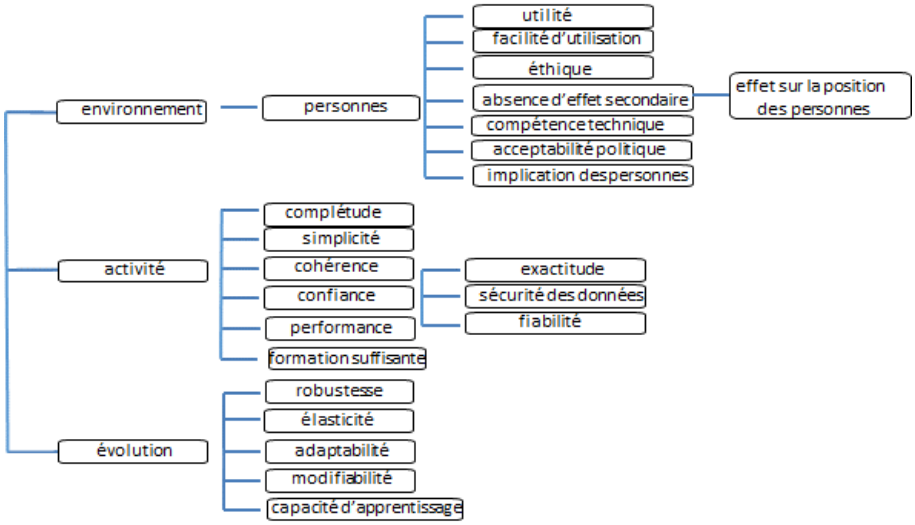


Figure 3. Hiérarchie adaptée aux groupes fonctionnels

La dimension *activité* d'un système renvoie à l'ensemble des processus qui s'y déroulent. Les principaux processus de la gouvernance de l'information sont : 1) la gouvernance des informations (planification, supervision et contrôle de la gestion

des informations et de leur utilisation) ; 2) la gestion de l'architecture des informations qui fait partie de l'architecture d'entreprise ; 3) la gestion de la sécurité des données et des informations inclus la protection de la vie privée, la garantie de la confidentialité et l'accès fiable et approprié pour l'utilisateur ; 4) la gestion des données de référence ; 5) la gestion des données pour l'analyse décisionnelle ; 6) la gestion des documents ; 7) la gestion des métadonnées ; 8) la gestion de la qualité des données. Une analyse de ces différents processus montre que cela relève davantage de la responsabilité de l'équipe de gouvernance de l'information que du management de l'entreprise. Il en est de même de la dimension *évolution*. La hiérarchie qui découle de cette analyse est représentée dans la figure 2. Elle représente une hiérarchie de critères permettant d'évaluer le succès de la gouvernance de l'information du point de vue du management.

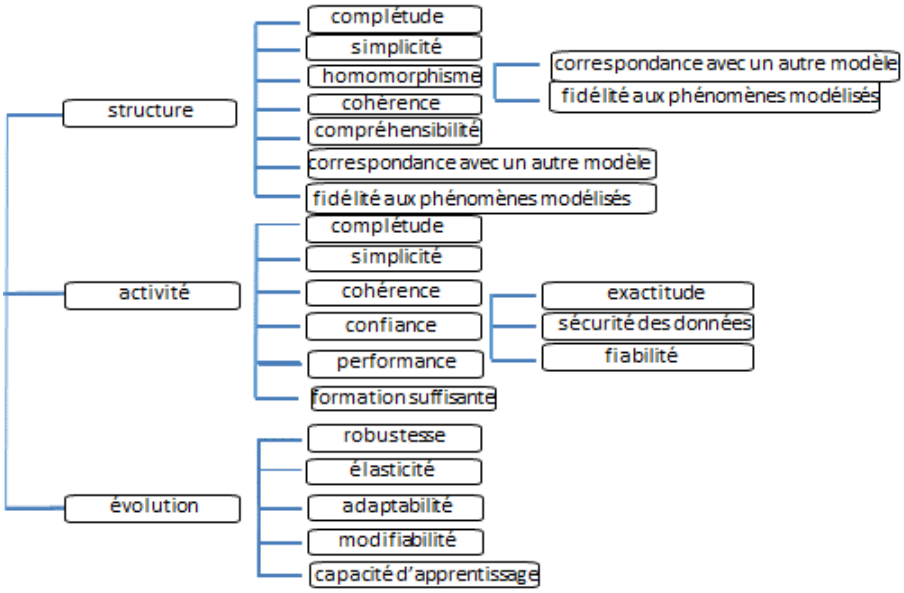


Figure 4. Hiérarchie adaptée au groupe GI

b. Hiérarchie de critères spécifique aux groupes fonctionnels

Rappelons que nous entendons par groupes fonctionnels (ou entités métiers) l'ensemble des acteurs appelés à mettre en œuvre la politique de gouvernance de l'information. Les groupes fonctionnels tels que ceux de la finance et du marketing sont intéressés, en premier lieu, par l'aide que peut apporter une politique de gouvernance dans le processus de décision. La qualité de ce dernier et l'effet sur la capacité du décideur à prendre des décisions sont des critères importants pour l'évaluation de la gouvernance de l'information. Ainsi, les préoccupations générales des groupes fonctionnels s'articulent autour de la capacité de la politique de gouvernance à faciliter les tâches de décision en termes de temps, de qualité et de flexibilité. Les dimensions principales que nous retenons sont *l'environnement*

(aspect *personnes*) ainsi que les dimensions *activité* et *évolution*. La figure 3 présente la hiérarchie de critères permettant de mesurer le succès de la politique de gouvernance du point de vue des groupes fonctionnels.

c. Hiérarchie spécifique au groupe Gouvernance de l'information

Trois dimensions sont essentielles pour cette partie prenante : s*tructure*, *activité* et *évolution*. Elles représentent la raison d'être de ce groupe de personnes. Ce dernier est, en effet, chargé d'élaborer la politique de gouvernance, de l'organiser et de la mettre en œuvre. La hiérarchie résultante est présentée dans la figure 4.

Sans prétendre prouver la complétude des différentes visions, elles résultent néanmoins de la spécialisation de la hiérarchie générique, ce qui leur confère une certaine homogénéité. Pour des raisons d'espace, nous ne pouvons pas détailler la hiérarchie pour la DSI qui contient principalement la faisabilité technique et la dimension Technologie de l'environnement.

3.4. La procédure d'évaluation

La procédure d'évaluation de chaque hiérarchie est fondée sur l'application de l'analyse hiérarchique multicritères (Saaty, 1994). Elle est représentée dans la figure 5 où :

- D représente un domaine à évaluer, par exemple « *objectif atteint* ». C'est donc un nœud non terminal de l'arborescence.
- SD représente un sous domaine, par exemple « *compatibilité technologique*» qui est le sous domaine de « *technologie* ».
- T représente l'évaluation d'un nœud terminal dont le poids est W et le résultat (ou la note) obtenu lors de l'évaluation est N. Ainsi, le couple ($W_{1,1}$; $N_{1,1}$) représente le poids et l'évaluation (la note) du nœud $T_{1,1}$.
- Les évaluations ne sont effectuées que pour les nœuds terminaux de la hiérarchie. Pour calculer la valeur de l'évaluation d'un sous domaine ou d'un domaine, il suffit de calculer la somme pondérée des nœuds terminaux associés au domaine ou au sous-domaine. Ainsi, pour le domaine D_1 dont le poids est W_1, la note obtenue est égale à $W_{1,1}$ * $N_{1,1}$ + $W_{1,2}$ * $N_{1,2}$. On procède ainsi en remontant la hiérarchie jusqu'à la racine.

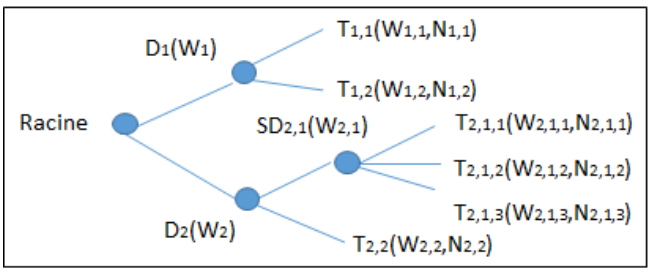


Figure 5. Analyse hiérarchique multicritères

La procédure d'évaluation comprend donc les phases suivantes :

- Pour chaque nœud de la hiérarchie, à l'exclusion de la racine, l'évaluateur indique l'importance de chaque critère. Cela peut être fait sur une base purement subjective en langage naturel. Ces pondérations sont ensuite transformées en des poids prenant des valeurs numériques, en s'assurant que la somme des poids des critères de chaque niveau de la hiérarchie est égale à 1.
- Nous procédons ensuite à l'évaluation proprement dite de chaque critère se trouvant sur les nœuds terminaux de la hiérarchie. Certains critères sont évalués de manière quantitative, essentiellement numérique. D'autres critères font l'objet d'une évaluation qualitative. Celle-ci est transformée en évaluation numérique en appliquant une échelle de correspondance.
- Puis nous évaluons la valeur de tous les autres nœuds inclus la racine de l'arborescence en procédant au calcul de la somme pondérée.

Bien que cela ne soit pas l'objectif de l'approche multifacette, la consolidation des vues, si elle est souhaitée par l'entreprise, est aisée à obtenir. Il suffit d'affecter un poids relatif à chaque partie prenante, d'y associer l'évaluation globale obtenue pour chacune d'entre elles et d'appliquer la procédure d'évaluation décrite plus haut. Il est possible d'envisager une évaluation semi-automatique du fait de la structure et des propriétés de l'arborescence hiérarchique multi critères. Toutefois, la transformation des évaluations qualitatives en valeurs numériques nécessite un traitement du langage naturel, ce qui rend l'opération plus difficile.

4. Application

Il s'agit d'une entreprise européenne de chimie de plus de 14000 employés. Ses produits améliorent la qualité, les fonctionnalités et l'apparence des plastiques. Ses marchés se répartissent dans 120 pays avec une présence forte en Europe et en Amérique du Nord. Le secteur d'activité est très compétitif du fait du coût de l'énergie et des matières premières. Une pression importante s'exerce sur les prix de ventes. Ceci nécessite une politique de support, en particulier à ces activités de vente.

L'entreprise a créé un concept de définition de produit pour décrire la composition de celui-ci en fonction des substances utilisées, de leur pays d'origine, afin de garantir la conformité aux exigences légales et réglementaires ainsi que la traçabilité et l'auditabilité du produit tout au long de son cycle de vie.

La stratégie de l'entreprise est fondée sur l'excellence opérationnelle. Les objectifs détaillés sont au nombre de quatre : optimisation des coûts, amélioration de l'efficience, accroissement de la transparence et renforcement de la profitabilité. Le concept de produit, bien que très original et très élaboré, n'a pas été contrôlé par un processus de management des données maîtres (MDM). Un audit de cette situation a fait ressortir plusieurs éléments : 1) une mauvaise qualité des données maîtres, 2) une incapacité à faire face aux enjeux et aux exigences métiers, 3) un manque de

traçabilité des modifications de données, 4) un niveau important d'activités manuelles consommatrices de temps pour nettoyer les données, 5) un niveau élevé de duplication des données maîtres, 6) des données manquantes, 7) une difficulté pour trouver les bonnes informations pour la décision.

Sur la base de cet audit, nous avons procédé à l'évaluation de la gouvernance de ces données maîtres à l'aide de la hiérarchie dédiée au management. Nous présentons ci-dessous notre analyse de cette situation en mentionnant entre parenthèses le critère correspondant. Le but d'excellence opérationnelle ne semble pas atteint du fait que la gouvernance des données maîtres ne permet pas aux décideurs de réaliser les tâches qui leur incombent (*réponse aux questions critiques*). Les éléments dont nous disposons sur la qualité des données nous amènent à conclure qu'en l'état, les données ne permettent pas d'assurer une décision de qualité (*qualité de la décision*). L'excellence opérationnelle ciblée par l'entreprise n'est pas atteinte totalement (*efficacité*). Les problèmes de qualité mentionnés lors de l'audit et leur impact sur la prise de décision éloignent l'entreprise de son but (*validité*). Nous ne disposons pas d'information sur le coût de la gouvernance et, par là-même, nous ne pouvons évaluer ni l'*utilité* ni la *faisabilité économique*. Il semble, de plus, que les différentes parties prenantes n'ont pas intégré la gouvernance des données maîtres dans leur pratique quotidienne (*faisabilité opérationnelle*). La politique définie dans cette entreprise est limitée aux données maîtres et ne couvre donc pas toutes les dimensions de la gouvernance de l'information (*généralité*). Malgré l'imperfection de la gouvernance mise en place, les données maîtres sont à la disposition des parties prenantes (*effet sur l'organisation*). En revanche, les problèmes mentionnés engendrent un dysfonctionnement des flux (*effet sur les flux d'information*). La mise en place de l'audit traduit une prise de conscience tardive du management (*implication du management*). A l'inverse, le niveau modéré de gouvernance a l'avantage de ne pas générer d'efforts de bord (*absence d'effets secondaires*). Une pleine exploitation de notre approche consisterait à enrichir cette analyse en affectant, d'une part, des poids aux différents critères et, d'autre part, des notes traduisant l'évaluation qualitative relatée ci-dessus. Pour des raisons d'espace, nous ne pouvons pas présenter l'application de l'approche aux autres parties prenantes (équipe de gouvernance de l'information et groupes fonctionnels). Néanmoins, cet exercice d'application de notre approche nous a permis de mesurer la richesse des éléments à évaluer, laquelle conduit à un processus fastidieux nécessitant un effort conséquent tant au niveau de l'équipe d'audit qu'au sein des parties prenantes impliquées dans l'opération.

5. Conclusion et recherches futures

La gouvernance de l'information constitue un enjeu primordial pour les entreprises. Sa mise en œuvre est coûteuse et peut impacter positivement ou négativement les processus métiers de l'entreprise et sa profitabilité. Nous avons présenté dans cet article une démarche d'évaluation de cette gouvernance qui prend en compte différents points de vue (management de l'entreprise, équipe de gouvernance de l'information, entités métiers, etc.) d'une part et différents critères

(efficacité, efficience, fiabilité, performance, etc.) d'autre part. L'organisation des critères obéit à la théorie des systèmes. Un cas simplifié illustre la démarche.

Nos recherches futures porteront sur la validation de l'approche en l'appliquant dans des conditions réelles et dans plusieurs contextes. Des hiérarchies spécifiques pour d'autres parties prenantes (Direction des systèmes d'information, Services juridiques) et par secteur d'activité (secteur public, entreprise fournisseur de données, secteur bancaire, etc.) seront aussi étudiées. Un prototype de support de l'approche sera développé pour faciliter cette validation. Nous utiliserons ensuite cette hiérarchie pour construire une démarche structurée d'audit de la gouvernance de l'information pour les entreprises et organisations.

Remerciements. *Les auteurs remercient les membres de la Chaire Stratégie et Gouvernance de l'Information de l'ESSEC, au sein de laquelle cette recherche a été réalisée.*

Bibliographie

Adelman L. (1992), Evaluating decision support and expert systems. Wiley, 1992.

Aier S., Fischer C. (2011), Criteria of progress for information systems design theories, Information Systems and E-Business Management, vol. 9, n°1, p. 133-172.

Ballard C. et al. (2014), Information Governance Principles and Practices for a Big Data Landscape http://www.redbooks.ibm.com/redbooks/pdfs/sg248165.pdf, IBM Redbooks Publication

Bennett S. (2017), What is information governance and how does it differ from data governance? Governance Directions, vol. 69, n°8, p. 462.

Boukhayma K., ElManouar A. (2015), Evaluating Decision Support Systems : A Literature Review, 15th International Conference on Intelligent Systems DeSign and Applications (ISDA)

Cleven A., Gubler P., Hüner K.M. (2009), Design alternatives for the evaluation of design science research artifacts, Proc. of the 4th Int. Conf. on Design Science Research in Information Systems and Technology (DESRIST 2009). Philadelphia, PA: ACM, 1–8.

Cronholm S., Goldkuhl G. (2003), Strategies for Information Systems Evaluation - Six Generic Types, Electronic Journal of Information Systems Evaluation, vol. 6(2), 65-74.

Foster J., McLeod J., Nolin J., Greifeneder E. (2018), Data work in context: Value, risks, and governance, Journal of the Association for Information Science and Technology, vol. 69, n°12, p.1414-1427.

Gartner. (2019), Gartner IT glossary, https://www.gartner.com/it-glossary/information-governance

Gregor S. (2010), Building theory in a practical science. In Information systems foundations: the role of design science (Eds Gregor, S. and Hart, D.). Australian National University Press, Canberra, Australia, p. 51-74.

Gregor S., Hevner A.R. (2013), Positioning and Presenting Design Science Research for Maximum Impact, MIS Quarterly, vol. 37, n°2, p. 337-355.

Hevner A.R., March S.T., Park J., Ram S. (2004), Design Science in Information Systems Research, MIS Quarterly, vol. 28, n°1, p. 75–105.

Järvinen P. (2007), On reviewing of results in design research, Proceedings ECIS 2007, St. Gallen, Switzerland, p. 1388-1397.

Le Moigne J.-L. (2006), Modeling for Reasoning Socio-economic Behaviors. Cybernetics & Human Knowing, vol. 13, n°3-4, p. 9-26.

March S.T., Smith G.F. (1995), Design and Natural Science Research on Information Technology, Decision Support Systems, vol. 15, n°4, p. 251–266.

Maynard, S., Arnott, D.R., Burstein, F. (1999), "A Method For Multiple Criteria Evaluation of DSS In A Multiple-Constituency Environment", Proceedings of the Fifth International Conference of the International Society for Decision Support Systems, Melbourne.

Peffers K., Rothenberger M., Tuunanen T., Vaezi, R. (2012), Design science research evaluation. Proc. of the 7th Int. Conf. on Design Science Research in Information Systems and Technology (DESRIST 2012). Las Vegas: Springer Verlag, 2012, p. 381–397.

Prat N., Comyn-Wattiau I., Akoka J. (2015), A Taxonomy of Evaluation Methods for Information Systems Artifacts, Journal of Management Information Systems, vol. 32, n° 3, p. 229–267.

Proença D., Vieira R., Borbinha J. (2016), A Maturity Model for Information Governance, Proceedings 11th Iberian Conference on Information Systems and Technologies (CISTI).

Rhee C., Rao H. R. (2008), Evaluating Decision Support Systems. Handbook on Decision Support Systems 2. Springer. 2008.

Saaty, T.L. (1994), Highlights and Critical Points in the Theory and Application of the Analytic Hierarchy Process, European Journal of Operational Research, 74, 426-447.

Scriven M. (1967), The Methodology of Evaluation. In: R.W. Tyler, R M. Gagne, M. Scriven (eds.), Perspectives of curriculum evaluation, Chicago, IL: Rand McNally, p. 39-83.

Simon H. (1996), The sciences of the artificial. 3rd edition. The MIT Press, Cambridge, MA.

Skyttner L. (2005), General systems theory: problems, perspectives, practice. 2nd edition. World Scientific, Singapore.

Sojda R. S. (2004), Empirical Evaluation of Decision Support Systems: Concepts and an Example for Trumpeter Swan Management. 2004.

Smallwood R. F. (2014), Information Governance: Concepts, Strategies, and Best Practices, London: Wiley.

Sonnenberg C., vom Brocke J. (2012), Evaluations in the science of the artificial: Reconsidering the build-evaluate pattern in design science research. Proc. of the 7th Int. Conf. on Design Science Research in Information Systems and Technology (DESRIST 2012). Las Vegas: Springer Verlag, 2012, p. 381–397.

Venable J., Pries-Heje J., Baskerville R. (2012), A comprehensive framework for evaluation in design science research. Proc. of the 7th Int. Conf. on Design Science Research in Information Systems and Technology (DESRIST 2012). Las Vegas: Springer Verlag, 2012, p. 381–397.

Caractérisation du temps de travail et de la durée nécessaires pour élaborer un logiciel de saisie et de gestion de données dans un laboratoire de recherche

Eric Quinton [1]

IRSTEA - Unité de recherche Écosystèmes aquatiques et changements globaux
50, avenue de Verdun
33612 CESTAS, France
eric.quinton@irstea.fr

RÉSUMÉ. *Pour faciliter la saisie et la gestion des données de recherche, le laboratoire Écosystèmes Aquatiques et Changements Globaux d'Irstea a développé plusieurs logiciels ces cinq dernières années. Dix d'entre eux ont été étudiés pour répondre à deux questions : est-il possible de déterminer le temps de développement nécessaire à partir d'un indicateur simple, et quel délai faut-il prévoir avant que le logiciel puisse être considéré comme achevé ?*

Le temps de développement peut être corrélé au nombre de tables de la base de données relationnelle sous-jacente. Quant au délai nécessaire pour achever un logiciel, il s'établit aux alentours de 22 mois à partir de la mise en production de la première version, pour tenir compte des évolutions liées à la prise en main de celui-ci. Cela implique de maintenir des ressources disponibles pendant toute cette période pour assurer la réussite du projet.

ABSTRACT. *To facilitate the input and management of research data, the Irstea Aquatic Ecosystems and Global Changes laboratory has developed several software packages over the last five years. Ten of them were studied to answer two questions: is it possible to determine the development time required from a simple indicator, and how long should it take before the software can be considered as complete?*

The development time can be correlated with the number of tables in the underlying relational database. As for the time required to complete a software, it is established around 22 months from the start of production of the first version. This time is necessary to take into account changes induced by the handling of it. It implies to maintain available resources throughout this period to ensure a successful project.

MOTS-CLÉS : *logiciel, développement, maturité, coût*

KEYWORDS: *software, conception, maturity, cost*

1. Introduction

Pour mener leurs travaux, les laboratoires de recherche peuvent organiser des campagnes de collecte de données, que celles-ci soient physiques (récolte d'échantillons) ou immatérielles (données issues de capteurs, observations, interviews, etc.). Le stockage de celles-ci dans les environnements numériques s'appuie de plus en plus sur les systèmes de gestion de bases de données.

Les saisies manuelles sont souvent fastidieuses et sujettes à de nombreuses erreurs. Dans les feuilles de calcul, on estime que 50 % d'entre elles contiennent des erreurs, et que le nombre de cellules erronées est de l'ordre de 1 à 2 % (Panko, 2008). De plus, l'absence de contrôle des types de données amène souvent à mélanger des données numériques avec du texte dans la même colonne, soit involontairement (mauvaise codification d'une date), soit volontairement pour exprimer une préoccupation ou remplacer une valeur discrète par un intervalle. L'exploitation de ces informations avec des outils de traitement automatique n'en est que plus difficile.

Le recours à des logiciels de saisie dédiés, qui intègrent des contrôles de cohérence et garantissent le typage des données, est souvent une solution judicieuse, dès lors que la collecte est répétitive (campagnes de récolte étalées sur plusieurs mois ou années, par des opérateurs internes ou externes, etc.) et justifie le temps passé à leur conception. Ces logiciels sont le plus souvent développés avec des interfaces web, mais peuvent également être écrits pour fonctionner en mode autonome, dans des zones où le réseau Internet ne peut être garanti.

L'unité de recherche Écosystèmes Aquatiques et Changements Globaux d'Irstea a mis en place depuis plusieurs années une dizaine de logiciels dédiés à l'acquisition des données collectées ou à leur gestion. Le code de certains de ces logiciels a été ouvert et mis à la disposition de la communauté scientifique. Ils ont été réalisés par une seule personne (administrateur de bases de données, développeur d'applications, ancien administrateur de systèmes d'informations), qui s'est appuyée sur des méthodes de développement basées sur les principes du manifeste Agile (Beck *et al.*, 2001).

1.1. Durée de mise au point du logiciel

Dans le développement traditionnel ou cycle en V (Forsberg, Mooz, 1998), le cahier des charges est rédigé et transmis à l'équipe de développement. La vérification de la conformité du logiciel s'effectue en comparant les fonctionnalités produites avec celles demandées.

Dans le développement Agile, les spécifications du logiciel sont élaborées au fur et à mesure que les premiers modules sont livrés. Entre la livraison de la première version opérationnelle et le moment où le logiciel peut être considéré comme achevé, il peut s'écouler plusieurs mois. Ce délai est nécessaire pour que le « client » puisse le découvrir, le faire évoluer pour qu'il réponde parfaitement à ses besoins, et enfin se l'approprier pour pouvoir l'utiliser. Ce processus d'appropriation a été modélisé en

trois phases : la découverte de la technologie, l'exploration, l'évaluation et l'adaptation et enfin l'utilisation proprement dite (figure 1) (Mendoza *et al*., 2010).

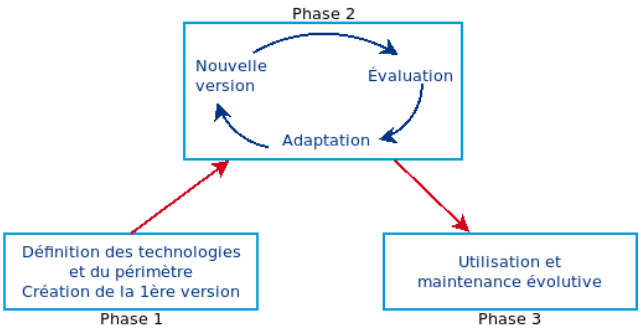


Figure 1. Les phases d'appropriation d'un logiciel (modifié d'après Mendoza et al.*). La première phase correspond à l'élaboration de la première version du logiciel, la seconde à son adaptation pour la rendre totalement utilisable et corriger les bogues de programmation. La dernière phase, dite de maintenance évolutive, est celle de son utilisation en routine.*

La phase de découverte permet de définir les contours techniques (application web ou client lourd embarqué, par exemple), mais également le périmètre de ce qui va être informatisé. Elle s'achève par la livraison de la première version utilisable.

La phase d'exploration est celle de l'adaptation. Elle comprend plusieurs étapes, comme le signalement des erreurs de programmation, la demande d'ajout de nouvelles fonctionnalités au fur et à mesure de la prise en main ou des évolutions ergonomiques. Compte-tenu du processus itératif de conception, elle peut s'étaler sur plusieurs mois, même si chaque version peut ne nécessiter que quelques jours de travail : les utilisateurs, en raison de leurs multiples activités et du délai nécessaire au processus cognitif d'appropriation, ont besoin de temps pour réaliser l'évaluation et exprimer leurs besoins complémentaires. C'est d'autant plus vrai dans les petites sociétés informatiques (Sánchez-Gordón, O'Connor, 2016), où répondre au besoin du client est vital, et où l'absence quasi-totale de cahier des charges impose une adaptation permanente. Le travail de développement dans un laboratoire de recherche peut s'apparenter à celui d'une petite société : il est rare de passer du temps à formaliser des processus s'il n'est pas nécessaire de le faire.

Enfin, la phase d'utilisation commence quand le logiciel est considéré comme mature. Il est alors utilisé en routine, et les évolutions, plus rares, sont dépendantes soit de la découverte tardive de bogues, soit de modifications dans les processus de gestion des données, soit de l'évolution des technologies utilisées (montées de version des *frameworks*, correctifs de sécurité, etc.).

La durée de la seconde phase est primordiale à connaître : même si elle nécessite peu de travaux, elle durera probablement plusieurs mois et est nécessaire pour mettre

au point le logiciel. Si les ressources ne sont pas disponibles pendant cette phase, le risque d'abandon du projet est grand faute de pouvoir l'adapter à la demande réelle.

1.2. Charge de travail

Un autre critère important avant de démarrer le développement d'un logiciel est celui de l'estimation de la charge de travail qui sera nécessaire.

Elle devrait être réalisée au plus tôt après le démarrage du projet, mais nécessite toutefois un certain recul. Pour la déterminer, deux approches sont en général utilisées : soit une estimation « à dire d'expert », soit l'utilisation d'une méthode de calcul formalisée. Jørgensen *et al*. (2009) ont estimé qu'il était probablement plus approprié de mixer les deux approches (même si l'une ou l'autre, dans certains cas, pouvait être plus adaptée), en utilisant une méthode de calcul et en la pondérant avec le regard d'un expert.

L'estimation du temps passé sur un projet de développement a largement été étudié. Papatheocharous *et al*. (2017) ont travaillé à la caractérisation des différentes phases (planification, développement, test, etc.). Al-Sabbagh et Gren (2018) se sont intéressés, quant à eux, à la relation entre la maturité de l'équipe et sa vitesse de développement. Si, depuis Boehm *et al*. (1995) et la création de la méthode COCOMO, le nombre de lignes produites reste une référence pour inférer le temps de réalisation, la manière de prédire ce nombre de lignes s'oriente de plus en plus vers l'attribution de points selon la complexité des tâches à réaliser, que ce soit à partir des cas d'utilisation produits en UML (Unified Model Language : http://www.uml.org/) (Clemmons, 2006 ; Usman *et al*., 2014), ou des *story points* (Coelho, Basu, 2012), largement utilisés lors des développements réalisés selon l'Extreme Programming (https://www.agilealliance.org/glossary/xp/). Sampaio *et al*. (2010) se sont intéressés, quant à eux, aux facteurs de productivité et aux stratégies à mettre en œuvre pour les maximiser et limiter les facteurs défavorables. En s'appuyant sur la théorie analytique de l'investissement dans le projet (Chen, 2006), Liu *et al*. (2015) ont démontré la relation dynamique entre la durée du projet, son niveau d'incertitude, son coût initial et sa productivité.

Tous les indicateurs proposés sont basés sur la quantification formelle des besoins : cas d'utilisations, *story points*, calcul du nombre de lignes, etc., et le recours à des métriques adaptées est un préalable obligatoire. Elles nécessitent une analyse complète des besoins selon certains formalismes (cas d'utilisation, par exemple). Cette étape est rarement réalisée pour des logiciels de petite taille.

Lors de la création d'une application de gestion de données, la conceptualisation de la base de données en est une des premières étapes. Le postulat est de considérer que les traitements prévus induisent la structure de la base de données sous-jacente, et que la structure de celle-ci est (ou sera) le reflet des processus informatisés. Nous avons choisi d'étudier s'il existait une relation entre la structure de la base de données et le temps passé.

Les bases de données relationnelles sont composées principalement de tables, qui sont reliées par des contraintes d'intégrité et sur lesquelles sont définis des index. Des méthodes de calcul de leur complexité ont été proposées (Calero *et al*., 2001 ; Pavlic *et al*., 2008). Elles s'appuient sur divers indicateurs comme le nombre d'attributs, le nombre de clés étrangères, le nombre d'index, etc. Toutefois, il n'est pas sûr que ces indicateurs soient pertinents pour définir le temps nécessaire à la réalisation d'un logiciel, ils renseignent surtout sur la complexité intrinsèque de celles-ci. Par contre, le nombre de tables, qui sont des regroupements logiques d'informations, pourrait permettre d'appréhender la complexité du logiciel. Celui-ci propose en général divers interfaces qui permettent de consulter ou de manipuler les informations qu'elles contiennent, et celles-ci sont en général conçues pour « coller » à la structure de la base de données sous-jacente (à une table correspond fréquemment une interface de saisie dédiée).

Lors des développements réalisés selon des méthodes agiles, l'ensemble des fonctionnalités qui seront fournies ne sont en général pas connues initialement. C'est au cours des différentes itérations qu'elles seront exprimées. Toutefois, au moment du lancement du projet, le commanditaire a déjà une idée assez précise de ce qu'il attend. Si des cas d'utilisation peuvent alors être décrits, ils reflètent en général assez mal la complexité sous-jacente. La nature des données manipulées, tant en entrée qu'en sortie, va influer fortement sur le codage nécessaire.

Notre postulat est de considérer que le nombre de tables nécessaires pour représenter et traiter l'information peut être un indicateur pertinent d'estimation du temps de réalisation d'un logiciel. En effet, entre une structure de données en deux dimensions (un tableau) et des structures plus complexes nécessitant de recourir à plusieurs tables pour les représenter, la charge de travail ne sera pas identique : des écrans de saisie supplémentaires seront nécessaires, des traitements de contrôle ajoutés, la documentation technique sera plus volumineuse, etc. Lors des études préalables, les données disponibles et celles qui sont attendues sont répertoriées, et c'est en fonction de celles-ci que la structure de la base de données va être construite, avant le codage proprement dit. Cette structure peut être issue d'une modélisation en classes (on ne retiendra alors que les classes persistantes qui pourront être couplées avec les tables de la base de données), ou directement depuis un modèle logique, qu'il soit complet (avec tous les attributs) ou non. Estimer la volumétrie du développement à partir du nombre de tables serait, dans ce contexte, assez simple à mettre en œuvre, et n'a guère fait l'objet de travaux à ce jour.

Nous avons cherché à répondre à deux questions :

– est-il possible d'estimer *a priori* le temps de travail qui sera nécessaire pour réaliser un logiciel en se basant sur le nombre de tables présentes dans la base de données relationnelle sous-jacente ?

– compte-tenu du processus de développement basé sur les principes du manifeste *Agile*, quelle durée faut-il prévoir pour qu'un logiciel soit considéré comme stable et ne fasse plus l'objet que d'opérations de maintenance techniques ou fonctionnelles ?

2. Méthodologie

2.1. Logiciels étudiés

En cinq ans, le laboratoire EABX a créé plusieurs logiciels permettant la saisie ou la gestion des données acquises sur le terrain. Nous en avons retenu dix suffisamment aboutis pour cette étude. La plupart ont été conçus pour fonctionner avec une interface Web et ont été écrits en PHP/HTML. Trois d'entre eux ont été écrits en Java avec une interface native écrite avec le composant *Swing* de Java (https://fr.wikipedia.org/wiki/Swing_(Java)), deux parce qu'ils devaient fonctionner en mode autonome (pas de connexion Web possible), et le dernier parce qu'il était destiné à des opérateurs et devait être facile à installer dans un poste de travail. La liste des logiciels retenus dans cette étude est décrite dans le tableau 1.

Tableau 1. Liste des logiciels créés

Nom	Description	Langage utilisé
Sturwild	Enregistrement des déclarations de captures accidentelles d'esturgeon d'Europe	PHP/HTML
Sturio	Gestion de la station d'élevage des esturgeons d'Europe	PHP/HTML
Sturatj	Enregistrement des captures scientifiques d'esturgeon d'Europe dans l'estuaire de la Gironde	Java
Pometweb	Enregistrement des relevés de pêche effectués dans le cadre du calcul de l'indicateur DCE pour les estuaires	PHP/HTML
Transect-php	Suivi des pêches scientifiques réalisées au droit de la centrale nucléaire du Blayais	PHP/HTML
TransectJ	Logiciel complémentaire du précédent, permettant de saisir les paramètres physico-chimiques et techniques de la pêche directement sur le bateau	Java
Otolithe	Logiciel de lecture des pièces calcifiées de poissons	PHP/HTML
Alisma	Saisie des relevés floristiques effectués dans le cadre de l'indicateur DCE pour les cours d'eau	Java
Collec-science	Enregistrement et gestion des échantillons scientifiques	PHP/HTML
Usact	Données d'enquêtes sociologiques concernant les conflits d'usage	PHP/HTML

Trois logiciels ont eu leur code source publié : Alisma (https://github.com/Irstea/alisma), Collec-Science (https://github.com/Irstea/collec) et Otolithe (https://github

.com/Irstea/otolithe). Les codes des autres programmes sont disponibles dans une forge interne à Irstea.

Les logiciels Web permettent d'alimenter une base de données relationnelle PostgreSQL (https://www.postgresql.org/), et ceux écrits en Java fonctionnent avec une base de données HSQLDB (http://hsqldb.org/), dont le moteur a la particularité d'être également écrit en Java et ne nécessite pas d'installer des outils complémentaires dans le poste de travail.

2.2. Données quantifiées

Le développeur a enregistré son temps de travail sur les différents projets en recourant au logiciel *Hamster Time Tracker* (https://github.com/projecthamster/). Il a également utilisé GIT (https://git-scm.com/) pour gérer son code, et a identifié les versions avec une numérotation à trois chiffres (Preston-Werner, 2013), qui permet d'identifier les versions majeures (numérotées sur un ou deux chiffres) des versions correctives (numérotées sur trois chiffres).

Le délai nécessaire à la mise au point du logiciel a été estimé en calculant le nombre de jours calendaires entre la première version mise en production et la version considérée *à dire d'expert* comme stable.

L'ensemble de ces données est récapitulé dans le tableau 2.

Tableau 2. Détails des logiciels créés

Nom	Nbre de lignes de code	Nbre de tables	Nbre de versions majeures	Nbre de versions totales	Nbre d'heures passées	Durée calendaire (en jours) de mise au point de la version stable
Sturwild	6811	21	2	7	139	680
Sturio	28587	103	8	31	491	685
Sturatj	17161	30	3	10	311	645
Pometweb	8401	22	6	9	171	679
Transect-php	6942	27	6	20	193	516
TransectJ	5623	9	3	7	102	454
Otolithe	7573	15	6	12	239	729
Alisma	16786	22	3	7	326	395
Collec-science	16898	31	6	23	596	701
Usact	10737	73	2	6	114	275

Le nombre de lignes de code ne prend en compte que le code spécifique à l'application, c'est à dire sans le code du *framework*. Il s'agit d'un nombre estimatif, qui peut différer selon les règles de formatage du texte implémentées dans les ou-

tils de conception (retour ou non à la ligne lors de la description de tableaux, règles de mise en forme des commentaires, etc.). Le *framework* PHP (https://github.com/equinton/prototypephp) est identique pour toutes les applications. Les logiciels Java ont été construits avec la même architecture technique sous-jacente. Le nombre de tables des bases de données qui sont alimentées par les logiciels a été comptabilisé, sans discriminer les tables de référence [1] des autres. Le nombre des versions a été calculé en recherchant les étiquettes (tag) positionnées dans le dépôt de code. Le nombre d'heures a été extrait des heures enregistrées dans *Hamster Time Tracker*. Ce temps inclut le développement, les réunions internes et la reprise éventuelle des données pour alimenter la base de données. Concernant le logiciel Collec-Science, celui-ci a fait l'apport de développements réalisés par des tiers pour intégrer de nouvelles fonctionnalités (gestion des méta-données, mécanisme de traduction pour supporter plusieurs langues notamment) ou améliorer certains aspects du logiciel. Le temps qu'ils y ont consacré n'a pas été intégré dans les calculs. Toutefois, le code réalisé est relativement minime en taille par rapport à l'ensemble de l'application, même s'il a pu être décisif dans la conception globale du produit.

Les logiciels ont été élaborés sur un laps de temps de 5 ans (figure 2). Le logiciel Alisma se distingue par une période d'écriture de la première version stable exceptionnellement longue comparée aux autres applications, en raison d'une indisponibilité des scientifiques chargés de réaliser les tests initiaux.

Enfin, les logiciels Otolithe, Alisma et Collec-Science sont diffusés en dehors du laboratoire et sont disponibles depuis Internet. Ils ont fait l'objet de travaux complémentaires, comme le dépôt à l'Agence de Protection des Programmes ou la rédaction de documentations techniques complémentaires, voire la création d'un site Web pour Collec-Science.

3. Résultats

3.1. Calcul de la charge de travail

Parmi les dix logiciels analysés, trois ont été développés en Java et sept en PHP/Html. Trois d'entre eux ont fait l'objet de travaux complémentaires pour être diffusés : Alisma (Java), Collec-Science et Otolithe (PHP).

Il est probable qu'un logiciel plus complexe nécessite plus de temps à être réaliser. Nous disposons de plusieurs métriques : le temps passé quantifié en heures, le nombre de lignes de code, le nombre de tables de la base de données sous-jacente et le nombre de versions produites.

1. Tables dont les valeurs évoluent peu et qui permettent d'alimenter les listes de sélection : taxons, sexe des animaux, etc.

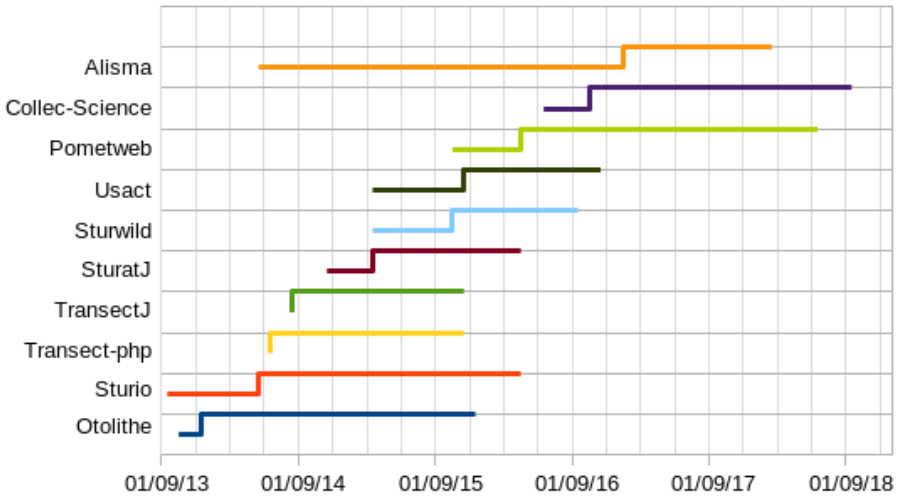


Figure 2. Calendrier de création des logiciels.
La partie basse de chaque ligne correspond à la période d'élaboration de la première version opérationnelle, la partie haute à la période de mise au point de la version stabilisée (cf. § 3.2). Le calendrier a été élaboré à partir du logiciel de gestion de code, et n'intègre pas les travaux préparatoires ou de création de la base de données précédant le début du codage.

3.1.1. Nombre d'heures rapportées au nombre de lignes de code

Nous avons d'abord analysé la vitesse de création du code, en calculant le nombre de lignes générées par heure (figure 3).

Pour la plupart des logiciels, la valeur est relativement proche (médiane de 51,5 lignes par heure pour les six logiciels aux centre du graphique).

Quatre logiciels ont des valeurs différentes. La productivité pour Usact est nettement plus importante (94 lignes par heure). Il s'agit d'un logiciel composé d'un nombre important de tables (73), mais qui sont toutes organisées sur le même modèle (règles de nommage et structuration identique des tables) : des *copier-coller* du code ou des opérations de factorisation ont permis d'accélérer largement le codage. Les travaux concernant Transect-php intègrent deux modules complexes : l'intégration des données issues de TransectJ, et la mise au point d'une interface de saisie adaptée à l'enregistrement des mesures réalisées en laboratoire. En ce qui concerne Otolithe, ce logiciel intègre un module écrit en Javascript qui permet de positionner des points sur une photo. La mise au point de ce module a été relativement longue, et justifie un temps passé plus important. Enfin, le logiciel Collec-Science intègre de nombreuses fonctionnalités avancées (lecture de codes-barres, gestion de méta-données, etc.) qui ont également été complexes à mettre au point. Ce logiciel a en outre été diffusé large-

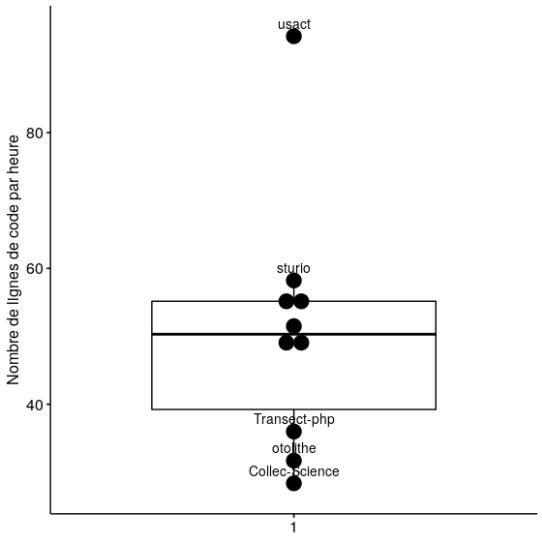


Figure 3. Nombre de lignes de code produites en une heure. La médiane vaut 51,5.

ment, et de nombreuses tâches annexes se sont rajoutées au temps de développement (rédaction de documentation technique, création d'un site Web, etc.).

Le nombre de lignes par heure peut être utilisé pour déterminer *a posteriori* si un logiciel a été plus ou moins complexe à produire, étant entendu que cette complexité peut être liée soit à un nombre de tâches annexes plus important, soit à des processus plus difficiles à coder.

3.1.2. Estimation du temps nécessaire à partir du nombre de tables

Le nombre de lignes de code rapporté au temps passé permet d'identifier les logiciels « classiques » de ceux qui sont plus consommateurs de temps. Mais l'estimation du nombre de lignes est complexe à mener, notamment dans les conceptions de type Agile : le dossier d'analyse n'est souvent que partiel et ne permet pas de disposer des métriques nécessaires pour ce calcul.

L'indicateur qui semble le plus facilement mobilisable pour estimer le temps nécessaire à la création d'un logiciel est le nombre de tables présentes dans la base de données. Nous avons vu précédemment que le temps nécessaire pour produire des lignes de code est assez constant d'un logiciel à l'autre.

La figure 4 permet de comparer le nombre de lignes produites par table et le nombre d'heures passées par table.

Elle permet de mettre en évidence plusieurs points.

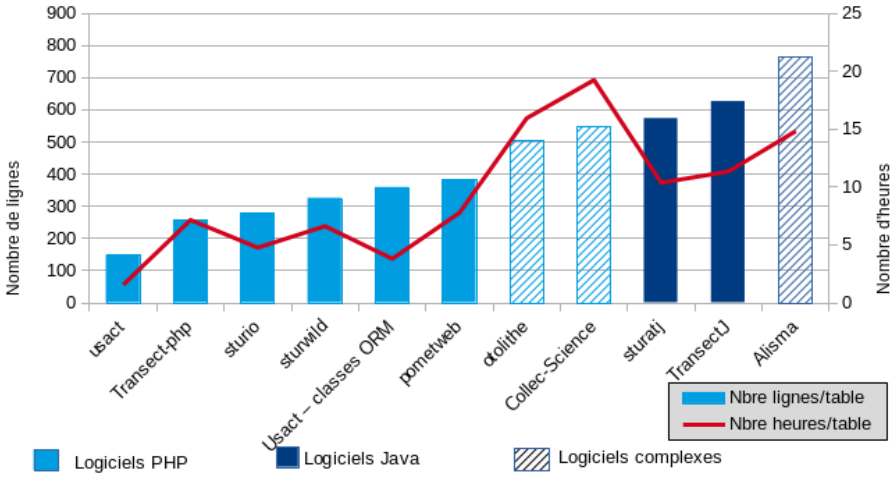


Figure 4. Nombre de lignes de code rapportées au nombre de tables.

D'une part, le nombre de lignes de code produit par table est plus important en Java qu'en Php (plus de 70 % de lignes en plus). Si trois logiciels se distinguent avec un nombre de lignes de code plus important (Collec-Science, Otolithe et Alisma), ils ont fait l'objet de fonctionnalités complémentaires, comme l'impression d'étiquettes et la lecture de codes-barres, la gestion de méta-données pour le premier, la gestion du positionnement de points sur des photos pour le second, et le calcul d'un indicateur par interrogation d'un service Web et l'ajout de fonctions d'exportation évoluées pour le troisième.

D'autre part, le temps nécessaire pour produire le logiciel rapporté au nombre de tables est assez constant pour un même langage de programmation. Les trois valeurs maximales (Otolithe, Collec-Science et Alisma) représentent l'effort nécessaire pour mettre au point les logiciels complexes, qui imposaient la mise place de solutions innovantes.

Le cas du logiciel Usact est particulier. S'il présente un nombre de lignes par table anormalement faible, c'est lié à la structure même de la base de données. Celle-ci est composée de 73 tables, mais la plupart d'entre elles ont des structures similaires. Dans les logiciels modernes, l'accès aux données stockées dans la base de données s'effectue par l'intermédiaire de classes dédiées qui encapsulent les différents accès (lecture, écriture, etc.). Elles sont souvent décrites sous le terme d'ORM : *Object Relational mapping*. Dans le cas du logiciel Usact, en raison de la conception générique du code produit pour gérer les tables de paramètres, le nombre de classes relevant du *mapping* de tables est de 30, soit une valeur significativement inférieure au nombre total de tables. En utilisant ce chiffre, le ratio du nombre de lignes par table se retrouve dans la moyenne des autres applications. Ainsi, si le nombre de tables est un indicateur facilement mobilisable, il ne semble être qu'une approximation du nombre de classes

ORM utilisées dans l'application. Ces deux chiffres sont très proches dans la plupart des cas, mais si les caractéristiques de la base de données et les méthodes de programmation l'imposent, il pourrait être utilement remplacé par le nombre de classes *ORM* générées.

Un biais dans ces estimations pourrait être lié à la capitalisation de l'expérience par le développeur : certains mécanismes mis au point pour un logiciel pourraient être utilisés dans d'autres, ce qui entraînerait un gain de temps lors de l'écriture des logiciels les plus récents. Après analyse, seul le logiciel Collec-Science a bénéficié de la mise au point de nouvelles fonctionnalités issues d'autres logiciels. On ne peut pas considérer que ce biais soit pertinent dans ce contexte.

À partir des éléments recueillis, il est possible d'envisager un indicateur simple de calcul du temps nécessaire, basé sur le nombre de tables (ou de classes *ORM* implémentées dans l'application) :

$$t = N \times ELC$$

ou *N* est le nombre de tables ou de classes *ORM*, et *ELC* un indicateur de complexité, composé de (E) : la vitesse intrinsèque de développement de l'équipe ou du développeur (expérience, gestion interne, organisation, etc.), (L) : du langage cible, et (C) : de la complexité intrinsèque liée du logiciel.

Les valeurs de E et de C sont difficiles à estimer. Il existe des indicateurs qui permettent de définir la complexité intrinsèque du code, comme l'indicateur « Cognitive complexity » (CC) (Campbell, 2018), qui compte notamment le nombre d'imbrications dans le code, et qui est calculé automatiquement notamment par SonarQube (https://sonarcloud.io). Nous avons cherché à savoir si la complexité ressentie, c'est à dire liée à la mise au point de nouveaux processus de traitement des informations, pouvait être corrélée à CC. Pour cela, les valeurs de E, L et C ont été empiriquement estimées, puis la valeur de CC a été calculée à partir de SonarQube. Comme il s'agit d'une valeur absolue, elle a été ramenée au nombre de tables de la base de données, pour permettre les comparaisons. Les résultats sont présentés dans le tableau 3.

La relation entre le facteur de complexité et le *cognitive complexity* divisé par le nombre de tables a été reportée dans la figure 5. Hormis pour Collec-Science, les valeurs sont très proches.

On peut ainsi considérer que l'estimation de E pour une équipe de développement pourrait être réalisée en se basant sur l'indicateur *Cognitive complexity* calculé sur ses productions précédentes, dès lors que l'on connaît le temps passé sur chaque projet et le langage utilisé.

L'approche présentée ici permet de définir, *a posteriori*, la vitesse intrinsèque de développement d'une équipe, résultat à la fois de son organisation, de ses compétences propres, de son implication, etc. Elle ne règle pas la difficulté d'estimation de la complexité du logiciel à écrire, qui est réalisée souvent de manière empirique à

Tableau 3. Simulation d'utilisation de l'indicateur sur le jeu de données
E : vitesse intrinsèque de développement, L : coefficient lié au langage, C : facteur de complexité, t : temps calculé,
CCt : Cognitive complexity divisé par le nombre de tables

Logiciel	Nb tables	Nb heures	E	L	C	t (en heures)	écart d'esti-mation du temps	CCt
Usact	73	114	7	1	0,2	102	-10 %	17
Transect-php	27	193	7	1	1	189	-2 %	90
Sturio	103	491	7	1	0,7	504	3 %	33
Sturwild	21	139	7	1	1	105	6 %	66
Usact-ORM	30	114	7	1	0,5	105	-8 %	42
Pometweb	22	171	7	1	1	154	-10 %	75
Otolithe	15	239	7	1	2	210	-12 %	145
Collec-Science	31	596	7	1	3	651	9 %	105
SturatJ	30	311	7	1,5	1	315	1 %	43
TransectJ	9	102	7	1,5	1	94	-7 %	44
Alisma	22	326	7	1,5	1,5	346	6 %	81

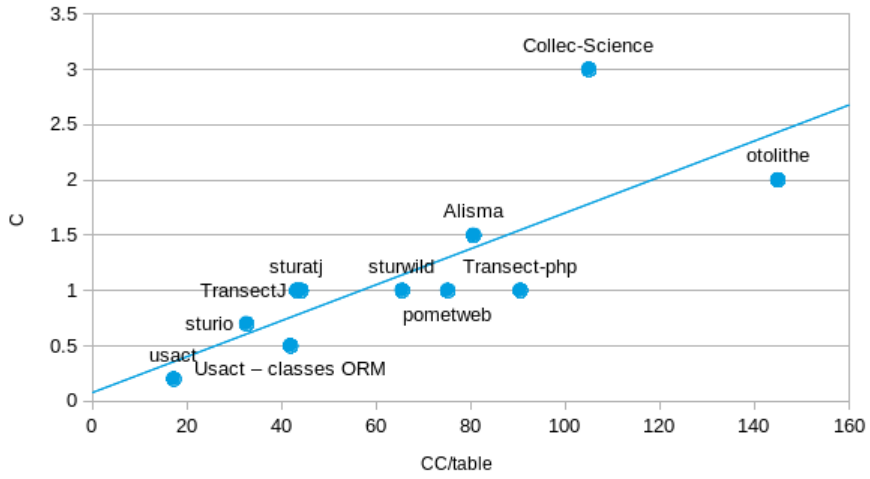


Figure 5. Relation entre le facteur de complexité (C) et le cognitive complexity divisé par le nombre de tables (CC/table)

« dire d'expert », ou en mobilisant des indicateurs basés sur des analyses plus fines, comme les *Story points* (Coelho, Basu, 2012), par exemple, s'ils sont disponibles.

L'autre inconnue, lors du démarrage d'un projet, tient à la difficulté à connaître la structure totale de la base de données, et notamment dans les développements agiles : sa structure est en général construite au fur et à mesure des *sprints* (en méthode SCRUM) (Morien, 2005). Dans la pratique, même si le détail de la base de données n'est pas encore connu dès le démarrage, la mise en place du projet implique d'en avoir une vision assez globale. Si le modèle détaillé, avec toutes les colonnes nécessaires, n'est pas encore élaboré, les principales entités (tables ou objets persistants, selon les méthodes d'analyse utilisées) sont définies lors des premiers travaux, et devraient être suffisantes pour quantifier le nombre de tables, au moins pour les besoins exprimés initialement.

Enfin, l'indicateur présenté ici ne peut fonctionner que pour les logiciels permettant de manipuler les informations stockées dans des bases relationnelles. Toutefois, compte-tenu du faible nombre de données, sa pertinence nécessiterait d'être testée dans d'autres contextes.

3.2. Calcul de la durée nécessaire pour qu'un logiciel arrive à maturité

Pour estimer la durée de mise au point d'un logiciel, la durée calendaire entre chaque version majeure a été étudiée (figure 6).

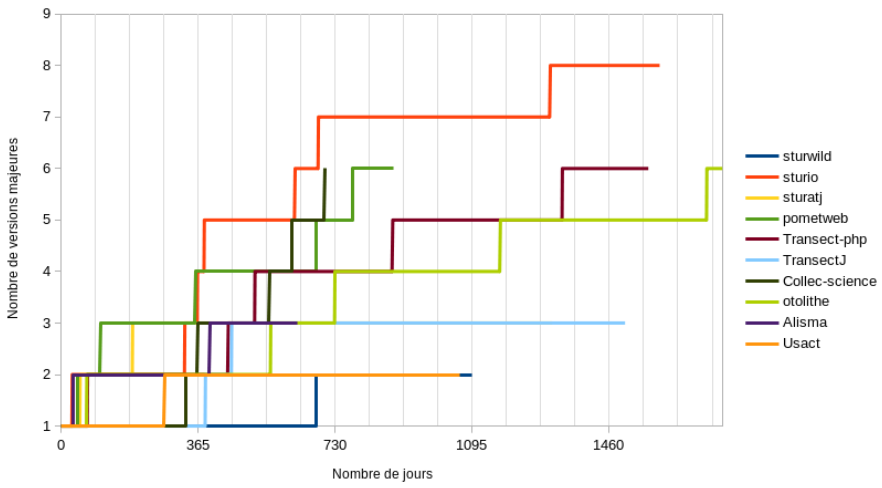


Figure 6. Durée entre chaque version majeure
Les courbes les plus longues correspondent aux logiciels les plus anciens. Dès lors qu'aucune nouvelle fonctionnalité n'est rajoutée (pas de nouvelle version majeure) pendant un laps de temps suffisamment important – environ 9 mois ici, le logiciel est considéré comme stabilisé.

Dès lors que la durée entre deux versions s'espace ou qu'une version majeure a une durée de vie approximativement supérieure ou égale à neuf mois, le logiciel est considéré comme stabilisé. Ce laps de temps a été déterminé *à dire d'expert*, en analysant chaque logiciel et l'ensemble des versions produites. Tous les logiciels ayant fait l'objet de modifications après la mise en production de la première version, aucun logiciel n'a été considéré comme stable avant la sortie de la seconde.

Cette durée s'établit entre 275 et 729 jours, pour une moyenne de 609 jours (20 mois) et une médiane de 662 jours (22 mois) (figure 7).

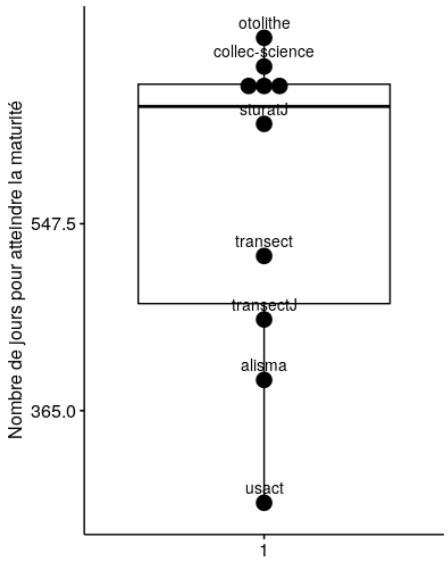


Figure 7. Nombre de jours nécessaires pour qu'un logiciel puisse être considéré comme mature.

Les logiciels Usact et Alisma se distinguent par des durées sensiblement plus faibles par rapport aux autres. Cela peut s'expliquer par le fait qu'ils ont été conçus à partir de logiciels déjà existants qui nécessitaient une réécriture (code non fonctionnel ou écrit avec des technologies obsolètes principalement) : les besoins étaient déjà connus et codés dans les outils précédents, et les utilisateurs n'ont eu besoin que d'évolutions limitées après la mise en production de la première version. Le logiciel TransectJ se caractérise par sa simplicité : il ne sert qu'à l'enregistrement de paramètres physico-chimiques lors d'opérations de pêche (un seul masque de saisie).

Six logiciels ont des valeurs très proches. En ne tenant pas compte des trois plus faibles valeurs qui s'expliquent assez facilement, l'écart n'est que de 213 jours (7 mois) entre Transect-PHP et Otolithe. Il semble raisonnable de penser que la durée nécessaire pour qu'un logiciel arrive à maturité s'établisse dans une fourchette entre dix-huit mois et deux ans, hors logiciels très simples ou inscrits dans un processus de recodage. Cette durée ne semble pas dépendante de de la prise en main du logiciel.

Les différentes applications créées sont utilisées, pour la plupart, par des personnes différentes, il n'y a pas d'effet lié à l'apprentissage de l'ergonomie. Il s'agit bien du délai nécessaire à la phase d'appropriation, qui est indépendante également de l'équipe de développement.

3.3. Discussion quant à la qualité des données

Le travail mené est empirique. Les temps de développement sont basés sur un enregistrement « au fil de l'eau » des tâches réalisées. Il s'apparente largement aux relevés réalisés par Todorovsky (1997 ; 2014), qui a calculé la répartition de ses tâches universitaires tout au long de sa carrière, entre enseignement, travaux scientifiques, tâches administratives, etc.

Le nombre de logiciels étudiés est faible, et ne permet pas des vérifications statistiques sérieuses. Toutefois, les données ont une certaine cohérence. Les logiciels ont été réalisés sur un laps de temps relativement court (5 ans), sans évolution ou rupture technologique majeure. Ils ont été écrits par un seul développeur, ce qui limite les effets liés aux compétences disparates. Le temps de travail a été enregistré de la même manière pour l'ensemble des logiciels, ce qui permet une comparaison longitudinale. La plupart ont été conçus sur une période courte : les premières versions opérationnelles de tous les logiciels ont été produites en trois ans. Toutefois, il reste global par logiciel, les différentes tâches (analyse, codage, voire reprise des données, mise en production, formation, etc.) n'ayant pas été comptabilisées individuellement : on en est réduit à considérer que chaque tâche est proportionnellement identique quel que soit le logiciel, ce qui est probablement inexact dans certains cas.

Enfin, les variations dans les résultats obtenus peuvent être expliqués et semblent bien refléter la perception de la complexité de chaque logiciel.

4. Conclusions et perspectives

L'analyse du temps passé pour créer un logiciel s'est déroulée sur plusieurs axes. Le premier consistait à vérifier s'il existait une corrélation entre le temps de développement total et le nombre de lignes de code produites. Les valeurs de la plupart des logiciels sont très proches et indépendantes du langage utilisé. Si un logiciel a été codé plus rapidement (45 % plus vite), deux l'ont été plus lentement (écart de 75 %). Ces écarts s'expliquent par la complexité intrinsèque du logiciel : dans le premier cas, le code a largement été recopié d'un module à l'autre, la structure sous-jacente étant très proche. Dans le second cas, des travaux complémentaires ont été nécessaires pour mettre au point le code (positionnement de points sur une photo, gestion de codes 2D et de méta-données notamment). Ainsi, le calcul du nombre d'heures par ligne de code semble être un indicateur pertinent de la complexité d'un logiciel.

Les méthodes permettant d'estimer le temps global nécessaire pour coder une application basées sur le nombre de lignes de code ne sont pas remises en question par ces chiffres. Toutefois, leur réelle difficulté de mise en œuvre est liée à l'estimation *a*

priori du nombre de lignes nécessaires. Toutes nécessitent une phase d'analyse poussée, avec description soit des écrans à produire, soit des cas d'utilisation ou des *User Stories*. En l'absence de ces informations, elles sont difficilement utilisables.

La productivité, lors de la conception de logiciels, est liée à de nombreux facteurs. Plusieurs classifications ont été proposées (Liu *et al*., 2015). L'une se base sur trois composantes principales : le logiciel lui-même à produire (réutilisabilité exigée, taille, complexité), l'équipe de développement (expérience, motivation, management, etc.) et le projet lui-même (langage de programmation, implication du client, etc.) (Sampaio *et al*., 2010).

La difficulté, pour estimer ce temps dans les petites structures, tient au fait qu'il n'existe pas ou peu d'indicateurs mobilisables facilement pour réaliser le calcul. En nous basant sur le nombre de tables présentes dans la base de données, nous avons montré qu'une certaine relation existait entre celles-ci et le temps global de développement. Toutefois, le nombre de tables ne permet pas de répondre à tous les cas de figure. Il est possible que cette valeur soit une approximation du nombre de classes *ORM* utilisées dans l'application, et que ce soit ce nombre qui soit le plus pertinent. Un seul logiciel présentait un écart important entre ces deux chiffres : il serait intéressant de vérifier si cette assertion se vérifie avec d'autres exemples.

Nous avons également cherché à savoir quelle durée était nécessaire pour mettre au point un logiciel, c'est à dire à le rendre suffisamment adapté aux utilisateurs en terme de fonctionnalités et d'ergonomie pour qu'ils puissent l'utiliser. Pour des logiciels conçus *ex nihilo*, il faut compter environ entre dix-huit mois et deux ans après la mise en production de la première version pour que les besoins d'évolutions s'espacent et qu'ils puissent être considérés comme stabilisés. Les deux cas où cette valeur était sensiblement plus faible correspondaient à des logiciels qui ont été refaits, les versions initiales ayant servi de modèle (*i. e.* de cahier des charges).

Ce délai s'explique par la nécessité pour les utilisateurs de se projeter dans l'outil créé, et c'est en manipulant les premières versions qu'ils peuvent définir leurs besoins réels. Il est indépendant de l'apprentissage de l'ergonomie (les logiciels ciblaient des utilisateurs différents), mais correspond probablement à un processus cognitif d'appropriation des usages et d'adaptation des méthodes de travail à l'informatisation des tâches. Une fois que les tâches évoluent en raison de l'informatisation, le logiciel est amené à évoluer également pour répondre aux nouveaux processus. Cette approche est pertinente pour les développements de type Agile : en accompagnant les utilisateurs et en définissant avec eux leurs besoins au fur et à mesure de l'avancement du projet, ceux-ci sont capables d'exprimer ce qu'ils en attendent.

Un des corollaires est qu'un projet nécessite un délai d'environ deux années pour aboutir une fois la première version fournie : même si le temps de développement n'est pas aussi important (il peut y avoir des périodes « creuses » avant que les utilisateurs fassent remonter leurs demandes d'évolution), des ressources doivent rester disponibles pendant toute ce laps de temps pour répondre à la demande. Que le projet soit développé en interne ou sous-traité, il est important d'intégrer ce délai dans la charge

de travail des développeurs ou dans le contrat de sous-traitance. Ainsi, le recrutement ponctuel d'un développeur sur une période courte (trois à six mois par exemple), une pratique assez répandue dans les laboratoires de recherche, ne serait pas suffisant pour aboutir à un résultat satisfaisant. S'appuyer sur des équipes pérennes semble indispensable, qu'elles soient internes à l'organisme, mutualisées, ou définies par contrat. Dans ce dernier cas, des mécanismes d'appel d'offres pourraient s'appuyer sur la fourniture d'unités de main d'œuvre, à utiliser durant une période de deux années par exemple.

Les résultats obtenus en analysant dix logiciels de gestion de données produits sur une période de cinq années nécessiteraient d'être confrontés à d'autres pratiques, et l'échantillon agrandi à d'autres laboratoires. La tentative d'estimation de la charge de travail à partir du nombre de tables de la base de données semble prometteuse notamment par sa simplicité de mise en œuvre. Elle nécessiterait toutefois d'être corroborée par d'autres études.

5. Remerciements

L'auteur tient à remercier Patrick Lambert pour ses conseils précieux tant au niveau de l'analyse des données que de la mise en forme de cet article.

Bibliographie

Al-Sabbagh K. W., Gren L. (2018, janvier). The connections between group maturity, software development velocity, and planning effectiveness. *Journal of Software: Evolution and Process*, vol. 30, n° 1, p. e1896. Consulté sur https://doi.org/10.1002/smr.1896

Beck K., Beedle M., Bennekum A. van, Cockburn A., Cunningham W., Fowler M. *et al.* (2001). *Manifesto for agile software development.* Consulté sur http://www.agilemanifesto.org/

Boehm B., Clark B., Horowitz E., Westland C., Madachy R., Selby R. (1995, décembre). Cost models for future software life cycle processes: COCOMO 2.0. *Annals of Software Engineering*, vol. 1, n° 1, p. 57–94. Consulté sur https://doi.org/10.1007/BF02249046

Calero C., Piattini M., Genero M. (2001). Database Complexity Metrics. *4th International Conference on the Quality of Information and Communications Technology*, p. 7. Consulté sur http://ceur-ws.org/Vol-1284/paper9.pdf

Campbell G. A. (2018). Cognitive complexity: An overview and evaluation. In *Proceedings of the 2018 international conference on technical debt*, p. 57–58. New York, NY, USA, ACM. Consulté sur https://doi.org/10.1145/3194164.3194186

Chen J. (2006, octobre). An analytical theory of project investment: a comparison with real option theory. *International Journal of Managerial Finance*, vol. 2, n° 4, p. 354–363. Consulté sur https://doi.org/10.1108/17439130610705535

Clemmons R. (2006, 02). Project estimation with use case points. *CrossTalk, The Journal of Defense Software Engineering*, vol. 19.

Coelho E., Basu A. (2012, août). Effort Estimation in Agile Software Development using Story Points. *International Journal of Applied Information Systems*, vol. 3, n° 7, p. 7–10. Consulté sur http://research.ijais.org/volume3/number7/ijais12-450574.pdf

Forsberg K., Mooz H. (1998, juillet). 7.17. System Engineering for Faster, Cheaper, Better. *INCOSE International Symposium*, vol. 8, n° 1, p. 917–927. Consulté sur https://doi.org/10.1002/j.2334-5837.1998.tb00130.x

Jørgensen M., Boehm B., Rifkin S. (2009, mars). Software Development Effort Estimation: Formal Models or Expert Judgment? *IEEE Software*, vol. 26, n° 2, p. 14–19. Consulté sur https://doi.org/10.1109/MS.2009.47

Liu L., Kong X., Chen J. (2015). How project duration, upfront costs and uncertainty interact and impact on software development productivity? A simulation approach. *International Journal of Agile Systems and Management*, vol. 8, n° 1, p. 39. Consulté sur https://doi.org/10.1504/IJASM.2015.068605

Mendoza A., Carroll J., Stern L. (2010). Software appropriation over time: from adoption to stabilization and beyond. *Australasian Journal of Information Systems*, vol. 16, n° 2. Consulté sur https//doi.org/10.3127/ajis.v16i2.507

Morien R. (2005). Agile development of the database: a focal entity prototyping approach. In *Agile development conference (adc'05)*, p. 103-110. Consulté sur https://doi.org/10.1109/ADC.2005.7

Panko R. R. (2008, février). Spreadsheet Errors: What We Know. What We Think We Can Do. *arXiv:0802.3457 [cs]*. Consulté sur http://arxiv.org/abs/0802.3457 (arXiv: 0802.3457)

Papatheocharous E., Bibi S., Stamelos I., Andreou A. S. (2017, octobre). An investigation of effort distribution among development phases: A four-stage progressive software cost estimation model. *Journal of Software: Evolution and Process*, vol. 29, n° 10, p. e1881. Consulté sur https://doi.org/10.1002/smr.1881

Pavlic M., Kaluza M., Vrcek N. (2008). *DATABASE COMPLEXITY MEASURING METHOD.* Consulté sur https://search.proquest.com/openview/72e3990c621d16ac14cde30b24bc5a0a/1

Preston-Werner T. (2013). *Semantic Versioning 2.0.0.* Consulté sur https://semver.org/

Sampaio S. C. d. B., Barros E. A., Aquino Junior G. S. d., Silva M. J. C. e., Meira S. R. d. L. (2010, août). A Review of Productivity Factors and Strategies on Software Development. In *2010 Fifth International Conference on Software Engineering Advances*, p. 196–204. Nice, France, IEEE. Consulté sur https://doi.org/10.1109/ICSEA.2010.37

Sánchez-Gordón M.-L., O'Connor R. V. (2016, septembre). Understanding the gap between software process practices and actual practice in very small companies. *Software Quality Journal*, vol. 24, n° 3, p. 549–570. Consulté sur https://doi.org/10.1007/s11219-015-9282-6

Todorovsky D. (1997, septembre). On the working time budget of the university teacher. *Scientometrics*, vol. 40, n° 1, p. 13–21. Consulté sur https://doi.org/10.1007/BF02459259

Todorovsky D. (2014, dec). Follow-up study: on the working time budget of a university teacher. 45 years self-observation. *Scientometrics*, vol. 101, n° 3, p. 2063–2070. Consulté sur https://doi.org/10.1007/s11192-014-1284-9

Usman M., Mendes E., Weidt F., Britto R. (2014). Effort estimation in agile software development: a systematic literature review. In *Proceedings of the 10th International Conference on Predictive Models in Software Engineering - PROMISE '14*, p. 82–91. Turin, Italy, ACM Press. Consulté sur https://doi.org/10.1145/2639490.2639503

Construction d'une méthode d'évaluation des actifs immatériels open source

Robert Viseur[1], Nicolas Jullien[2]

1. Faculté Warocqué (TIC), Université de Mons
robert.viseur@umons.ac.be

2. IMT Atlantique, Marsouin-LEGO
nicolas.jullien@imt-atlantique.fr

RÉSUMÉ. L'écosystème open source est marqué par un historique de rachats d'entreprises pour des montants parfois très élevés (p. ex. $34.10^9$$ pour Red Hat en 2018). Cette observation motive notre question de recherche : « comment estimer, dans les comptes d'une entreprise, la qualité et la valeur d'un actif immatériel ouvert de type logiciel open source ? ». L'approche proposée, basée sur une méthode documentée d'évaluation des actifs immatériels ainsi que sur un ensemble d'outils existants dédiés à l'évaluation des logiciels, notamment open source, propose une étape de notation et une étape d'estimation de la valeur. Quatre types de capital associés au logiciel open source sont ici pris en compte : le capital technique (code source), le capital communautaire (contributions), le capital marque et le capital client (performances financières). Nous proposons comme résultat une architecture d'évaluation ainsi qu'un premier exemple d'application de la méthode.

ABSTRACT. The open source ecosystem is marked by a history of corporate buybacks for sometimes very high amounts (eg $ 34.10^9$ for Red Hat in 2018). This observation motivates our research question: « how to estimate the quality and value of an open, open source software type intangible asset for a company? ». The proposed approach, based on a documented method for evaluating intangible assets as well as on a set of existing tools dedicated to the evaluation of software, including open source software, proposes a step of rating and a step of estimation of the value. Four types of capital associated with open source software are taken into account: technical capital (source code), community capital (contributions), brand capital and customer capital (financial performances). We propose an evaluation architecture as well as a first example of application of the method.

Mots-clés : open source, évaluation, capital immatériel, cocomo, qualité.

KEYWORDS: open source, assessment, intangible capital, cocomo, quality.

1. Introduction

Le concept d'*open innovation*, popularisé par les travaux de Chesbrough et al. (2006), a suscité l'intérêt tant des chercheurs que des industriels. Les entreprises y sont invitées à exploiter les innovations produites par des tierces parties (*inputs*) et à valoriser les innovations produites à l'intérieur de l'entreprise (*outputs*), pas seulement au travers de la commercialisation de nouveaux produits mais aussi en accordant des droits d'exploitation à des entreprises tierces (p. ex. licences). Ces transactions supposent typiquement l'échange de droits de propriété intellectuelle (p. ex. brevets). L'ouverture du processus d'innovation précède cependant les travaux de Chesbrough. Les décennies précédentes ont en effet vu l'émergence du logiciel libre et *open source* (voir l'Open Source Definition, https://opensource.org/osd) dans le domaine logiciel. Face à la réalité des essaimages tels que l'*open data* (données) ou *open hardware* (biens manufacturés), Pénin (2011) a proposé un cadre, appelé *open source innovation*, généralisant les principes de l'*open source* (logiciel) et se distinguant de l'*open innovation* par le large partage des connaissances (p. ex. code source), une organisation sans hiérarchie (développement) et une régime d'appropriabilité faible (licences). Cependant, West (2003) a montré que, même au sein des processus *open source*, des degrés d'ouverture différents étaient possibles (p. ex. approches *open parts* ou *partly open*). Cette question du degré d'ouverture se pose en dehors du secteur *open source* (p. ex. co-création avec les utilisateurs) et implique d'ailleurs des changements au niveau de l'usage des outils de propriété intellectuelle.

Dans le domaine logiciel, l'*open source* est l'objet de recherches depuis une quinzaine d'années, notamment sur les modèles d'affaires ou sur la gouvernance des projets. L'implication croissante des entreprises a également été analysée par Fitzgerald (2006). Au fil des années, beaucoup d'entreprises, jeunes pousses ou sociétés ayant pignon sur rue, ont choisi d'investir dans l'*open source* : Netscape (1998), IBM (2001), Novell (2003), Sun Microsystems (2006),... jusqu'au retournement retentissant de Microsoft positionnant dès 2012 sa plate-forme *cloud* Azure comme le plus grand *cloud open source* au monde ! Des rachats importants d'entreprises *open source* ont également eu lieu : MySQL ($1 milliard), Cygnus Solutions ($674 millions), JBoss ($350 millions) ou Zimbra ($350 millions). Le dernier en date concerne la société Red Hat, rachetée en 2018 par IBM pour 34 milliards de dollars. Ils posent en particulier la question des méthodologies permettant d'estimer la valeur d'une entreprise *open source* et des actifs associés. En effet, les méthodologies permettant d'évaluer la valeur d'un logiciel (p. ex. COCOMO et SQALE) ne suffisent pas, car elles ne mesurent, le plus souvent, que la valeur de recréation (le temps de travail pour refaire le logiciel à l'identique), et ne prennent pas en compte la dimension marchande (services, produits) du projet, qui est au cœur du travail que nous souhaitons mener. D'autre part, même pour évaluer l'effort à réaliser par l'entreprise étudiée, il faut aussi tenir compte du fait que la propriété du projet (p. ex. code source) est partagée avec d'autres contributeurs et que donc l'entreprise n'a pas forcément à tout redévelopper.

D'un point de vue méthodologique, l'intérêt croissant pour la question de l'évaluation des actifs immatériels ouverts (tels que les systèmes d'information, les logiciels et les marques) et les solutions qui ont été proposées (Bonnet, 2011 ; Fustec & Marois, 2006, 2016), rendent aujourd'hui possibles la réutilisation de certaines de

ces propositions pour la création d'une méthode outillée d'évaluation de projets *open source*.

2. Organisation de la recherche

Cet article présente un travail préliminaire dont l'objectif est de dessiner une forme de micro-évaluation d'actifs immatériels ouverts. Il peut être vu comme une proposition de développement d'une méthode simplifiée, permettant une estimation rapide de la valeur et de la qualité d'un actif, qui peut, ensuite être complétée par une approche plus précise, plus complète mais aussi plus complexe. Si le problème adressé est multi-factoriel, le parti-pris adopté pour sa résolution consiste à réutiliser et intégrer des méthodologies et des outils existants en matière d'évaluation des logiciels. Dans ce premier essai, le type d'actif considéré est un logiciel *open source*.

Nous chercherons à apporter une première réponse à la question suivante : « *comment estimer la qualité et la valeur d'un actif immatériel ouvert de type logiciel open source pour une entreprise qui contrôle tout ou partie de cet actif ?* ». Notre article sera organisé en quatre parties. Dans la première, nous présentons une méthode d'évaluation des actifs immatériels. Dans la seconde, nous discutons des spécificités des logiciels *open source* et faisons une courte synthèse sur les outils existants en matière d'évaluation de tels logiciels. Dans la troisième, nous présentons la démarche d'évaluation proposée et l'appliquons sur un exemple. Dans une quatrième et dernière partie, nous discutons les résultats de la recherche et en précisons en particulier les limitations actuelles ainsi que les pistes d'amélioration identifiées.

3. Évaluation des actifs « immatériels » (gazeux)

3.1. Principe

Fustec et Marois (2006, 2016) ont publié une méthode d'évaluation des actifs qu'ils appellent « immatériels ». Ils y distinguent ainsi les richesses dites solides (immobilisations, actifs matériels, immatériels et financiers sur le plan comptable), liquides (actifs circulants, p. ex. stocks) et gazeuses (qui sont nommés actifs immatériels, mais qui correspondent plutôt au *goodwill* sur le plan comptable). La norme internationale IAS-IFRS permet la valorisation sous forme de *goodwill*, ou « écart d'acquisition », pour les marques, les savoir-faire et la R&D, les possessions artistiques, les fichiers clients, les contrats et les licences ainsi que le système d'information[1]. Ces actifs « gazeux » de Fustec et Marois peuvent être valorisés dès lors qu'ils font l'objet d'une acquisition, dont l'évaluation peut uniquement bénéficier d'une révision à la baisse. Ils permettent d'expliquer, sous cette forme d'un *goodwill*, l'écart entre le coût d'acquisition et la valeur des actifs identifiables (p. ex. machines, brevets, titres de propriétés) d'une entreprise.

[1] L'approche présentée dans cet article porte sur la notation et l'évaluation d'un projet logiciel, pas de sa mise en œuvre au sein d'un système d'information. Elle devrait être complétée pour l'évaluation d'un système d'information d'entreprise dans lequel va par exemple être prise en compte la capacité des acteurs métiers à influencer les données, les règles et les processus mis en œuvre. Dans cette vision développée notamment par Pierre Bonnet (2011), un logiciel développé dans les règles de l'art mais totalement contrôlé par les développeurs souffrira d'une mauvaise notation.

On retrouve, chez Fustec et Marois (2006) les mêmes types d'actifs gazeux, qu'ils séparent en deux catégories : d'une part, le capital client, d'autre part, les autres, incluant le capital humain, le capital organisationnel, le système d'information, le capital savoir, les marques, le capital partenaire, le capital actionnaire et le capital environnemental. En effet, les clients constituent une source de *cash* (et des collecteurs, ou utilisateurs de produits) tandis que les autres sont des collecteurs, ou utilisateurs de *cash* (et des sources de produits). Un capital client plus faible peut indiquer une mauvaise performance des actifs gazeux consommateurs de *cash*, l'inverse pouvant constituer un indice de l'épuisement progressif de ces actifs.

La suite de la méthode décrite par Fustec et Marois (2006) distingue la notation et le calcul de valeur de tels actifs gazeux, qui sont donc à la base du calcul du *goodwill* que nous proposons, dans le cas d'un logiciel *open source*.

3.2. Notation

La notation passe par la construction d'un arbre composé de critères et de sous-critères mesurables. La notation des différents actifs immatériels est réalisée sur une échelle de « A » à « F ». Ainsi, un actif noté « A » est de très bonne qualité au contraire d'un actif noté « F ». Un système de conversion permet de transformer la valeur mesurée en note (Fustec & Marois, 2006). Chaque actif fait l'objet d'une décomposition analytique en critères, dont la pondération peut être adaptée pour les besoins de l'évaluateur.

3.3. Estimation de la valeur des actifs

L'évaluation des actifs peut être réalisée de différentes manières par la valeur de re-création (notée V-CRE), par la valeur de rendement (notée V-REND) ou par la comparaison à des transactions comparables (notée V-COMP) .

Le calcul de la valeur de rendement se fait sur base du *cash flow* actualisé cumulé (noté DCF pour *Discounted Cash Flow*). Cette approche est en particulier retenue pour l'estimation de la valeur du capital client. La pertinence de ce type de méthode d'évaluation financière est critiquée pour l'évaluation des *start-ups* qui reposent davantage sur une promesse d'activité et présentent un risque d'échec élevé (Miloud & Cabrol, 2011). Une approche similaire portant sur un projet particulier peut être mise en œuvre en calculant la Valeur Actuelle Nette du projet sur l'horizon de temps considéré (Fraix, 1998).

4. Évaluation des projets *open source*

4.1. Principe

L'évaluation d'un projet open source doit comporter un volet « notation » et un volet « estimation de la valeur », conformément à la méthodologie présentée par Fustec et Marois (2006, 2016).

Quatre types de capital doivent être distingués, puis évalués : le capital technique, le capital communautaire, le capital marque et le capital client. Les trois premiers constituent une forme d'actifs consommateurs de cash au sens de Fustec et Marois (2006). Le capital technique recouvre le logiciel proprement dit (code source). Le capital communautaire recouvre les apports de la communauté. Dans le

cadre de cet article, afin d'en faciliter l'évaluation, cet apport sera limité aux contributions en code source. Le capital marque recouvre la marque associée au projet. Enfin, le capital client représente les clients possédés par l'entreprise, ou ceux qu'elle peut espérer conquérir en s'appuyant sur une valorisation marchande du logiciel *open source*. C'est sans doute la partie la plus compliquée à appréhender par rapport à un calcul de *goodwill* classique, et aussi celle qui détermine le modèle de transformation des notes en valeur.

4.2. Modèles d'affaires open source

Si l'on prend le point de vue d'une entreprise qui utilise un ou des logiciels open source pour construire son offre commerciale, on constate que la capture de la valeur par les entreprises peut se faire en monétisant l'accès au logiciel ou en développant des activités de services.

Les opportunités de monétisation de l'accès dépendent du régime d'appropriabilité, généralement faible dans le cas des logiciels *open source* mais cependant modifiable par le biais d'outils juridiques comme les accords de contributeurs (Muselli, 2008 ; Viseur, 2013). Si les licences *open source* organisent le partage de la propriété du logiciel entre tous les contributeurs, l'accès au logiciel peut être monétisée par un modèle de double licence par le titulaire des droits (Viseur, 2013). Le logiciel est alors publié sous une licence *open source* et sous une forme propriétaire, avec ou sans différenciation technique, nécessitant le paiement d'une licence d'utilisation pour ce dernier. Par extension, l'éditeur peut également réglementer et monétiser l'utilisation de sa marque (Fitzgerald, 2006).

Les activités de service peuvent être associées, en utilisant la typologie de Jean Gadrey (2003) à l'entretien de capacités techniques (Gadrey, 2003) ou à de la fourniture d'expertise, de capacités humaines. Dans le premier cas, l'entreprise garantit, avec le soutien éventuel de sa communauté, l'évolution du logiciel face à celle de l'environnement légal et technologique. Elle se rémunère en monnayant l'accès au logiciel ou l'assurance de son bon fonctionnement. Dans le second cas, l'entreprise fournit de l'assistance (p. ex. formation et *helpdesk*) ou de l'adaptation (p. ex. développement d'un nouveau module d'ERP). Jullien et Zimmermann (2011) résument les différentes dimensions des services proposés sous l'abréviation « 3A » recouvrant l'Assurance, l'Assistance et l'Adaptation.

Charleux et Mione (2018) ont, elles, pris le point de vue d'un projet *open source*, pour s'interroger sur les modèles d'affaire que des entreprises pouvaient développer, ou plus exactement que ce projet pouvait induire. On retrouve, d'une autre manière, les différentes formes de valorisation indiquées dans le paragraphe précédent (nous citons) : l'optimisation, l'expertise, l'exploration et l'engagement.

L'optimisation correspond à une modèle d'édition commerciale où l'accès au logiciel est monétisé via un mécanisme de licence (p. ex. double licence et *open core*), ou via un mécanisme d'édition de versions vérifiées et garanties (p. ex. RedHat), ou via l'accès à une plate-forme qui met en œuvre le logiciel (modèle de *cloud computing*). Ce modèle est donc étroitement fondé sur l'entretien de capacités techniques. L'expertise recouvre un modèle de services, dédiés à l'utilisation du logiciel (p. ex. paramétrage et aide à l'utilisation) ou à l'amélioration de celui-ci (p. ex. développements complémentaires). Ce modèle est donc lié à l'entretien du capital humain et présente un potentiel de croissance plus limité. L'exploration

désigne le modèle des fondations permettant de mutualiser l'effort de développement dans le cadre d'une gouvernance plus neutre et ouverte. Ces logiciels produits sous le patronage des fondations constituent une forme de bien public industriel au sens de Romer (Jullien & Zimmermann, 2013). L'engagement recouvre pour sa part les projets *open source* développés au sein de communautés davantage informelles, qui nous intéresse moins ici, puisqu'il n'y a pas d'implication marchande, pour ces auteures.

Ce que montrent Jullien & Zimmermann (2011), s'appuyant sur les travaux de West (2003), ou de Dahlander & Wallin (2006), c'est que les modèles de valorisation marchande vont, eux-mêmes, comme pour tout logiciel, dépendre du type de logiciel et de ses usages, du nombre et des caractéristiques des utilisateurs actuels (sa part de marché) et aussi des utilisateurs potentiels (individus, indépendants, petites, moyennes, grosses entreprises, novices, experts).

Une stratégie *open source* se justifie s'il y a suffisamment d'utilisateurs experts, développeurs[2], pour créer une communauté, un projet *open source*, avec plusieurs développeurs (co-développement du logiciel), ou au moins des utilisateurs capables de signaler et éventuellement de corriger des erreurs, ou « *bugs* », et d'exprimer des spécifications de fonctionnalité[3].

Plus le projet de développement logiciel est dynamique, plus il y a de la place pour les activités de service autour du logiciel (p. ex. maintenance en condition opérationnelle, adaptation aux besoins et formation), mais aussi plus l'entreprise qui souhaite proposer ces services devra s'impliquer dans les développements (et l'animation de la communauté) pour pouvoir contrôler l'évolution du logiciel, et être capable de garantir à ses clients que leurs besoins seront pris en compte (Jullien & Zimmermann, 2009).

Les entreprises qui peuvent proposer des offres commerciales sont celles qui contrôleront l'accès, ou la dynamique du projet, donc qui s'impliquent dans la production du logiciel, et qui sont associées au logiciel (notamment en termes de marque ou d'expertise sur le logiciel). Nous pouvons distinguer trois formes de contrôle : le contrôle par la propriété intellectuelle (nécessaire pour contrôler l'accès dans le cas des doubles licences, moins dans le cas des offres de type *cloud*), le contrôle par les ressources et le contrôle par le *leadership*. (Fitzgerald, 2006 ; Schaarschmidt et al., 2015). Le contrôle de la propriété intellectuelle passe par une propriété accrue du code source (p. ex. accords de contributeur, mais qui risque de diminuer la dynamique communautaires) ou par la détention de propriété intellectuelle complémentaire (p. ex. marque) renforçant l'emprise sur le projet. Le contrôle par les ressources consiste à orienter le projet selon le principe que celui qui fait décide des orientations (p. ex. affectation de développeurs, qui coûtent chers). Le contrôle par le *leadership* passe notamment par l'embauche ou le *sponsoring* de personnalités clefs (p. ex. *committers* influents).

---

[2] Ce qu'ils appellent les utilisateurs « von Hippel », ou VH, en référence au modèle de l'utilisateur-innovateur, expert, qui développe les produits dont il a besoin, et qui est, d'après Von Hippel (2001), celui du logiciel libre.

[3] Ce qu'ils appellent les utilisateurs « Kogut-Metiu », ou KM, en référence à l'article de Kogut & Metiu (2001) qui signale que de 50 à 80 % du coût de développement d'un logiciel correspond à la correction des *bugs*, et que c'est une des principales raisons de l'efficacité du modèle *open source* (accès à une base large de testeurs-correcteurs).

Pour résumer, un projet *open source* (vision de Charleux et Mione, 2018) a d'autant plus de chances de se développer qu'il intéresse des utilisateurs (part de marché) et surtout des utilisateurs experts (communauté de développeurs). Ces utilisateurs vont choisir ce logiciel s'il répond à leurs attentes (donc qu'il est suffisamment mâture technologiquement, s'il a un TRL[4] suffisamment élevé). D'un autre côté, plus un logiciel sera dynamique en matière de développement (communauté active, mais aussi besoin de développer de nouvelles fonctionnalités), plus il y aura de la place pour des offres complémentaires marchandes.

Plus les entreprises marchandes qui proposent ces offres s'impliquent dans le développement du logiciel, plus elles peuvent le contrôler et en extraire une valeur marchande, mais cela a un coût (rémunération des développeurs) et risque, si le contrôle est trop fort, de diminuer la dynamique communautaire.

L'évaluation de la valeur d'un projet *open source* pour une entreprise passe donc par l'évaluation de la valeur du projet (en termes de développement, mais aussi de base utilisateur, de notoriété ou capital marque, notamment), puis de la part de cette valeur que l'entreprise peut capter, qui dépend en partie de son degré de contrôle (du projet, mais aussi des marques développées par celui-ci) et en partie des caractéristiques des utilisateurs. C'est ce que nous avons essayé de prendre en compte dans notre modèle, en nous appuyant, dès que possible, sur les outils d'analyse déjà développés.

4.3. Projets open source

L'évaluation des projets *open source* a fait l'objet de plusieurs tentatives de création de méthodologies standards. C'est notamment le cas d'OpenBRR et de QSOS (Deprez & Alexandre, 2008). QSOS (2013), publiée par Atos Origin, est probablement la plus populaire en France. Elle montre en particulier comment un outil d'évaluation, basée sur une échelle de mesure documentée, peut limiter la subjectivité des évaluateurs. À partir de 2010, des recherches pour automatiser les mesures de qualité des projets *open source* développés dans un cadre communautaire ont par ailleurs été entreprises (Izquierdo-Cortazar et al., 2010).

5. Mise en œuvre de l'évaluation des actifs immatériels *open source*

5.1. Notation

Capital technique

Le capital technique peut être noté à partir de la qualité du logiciel et de la qualité du processus de production du logiciel.

[4] Pour « *Technology Readiness Level* » : voir Mankins (2009).

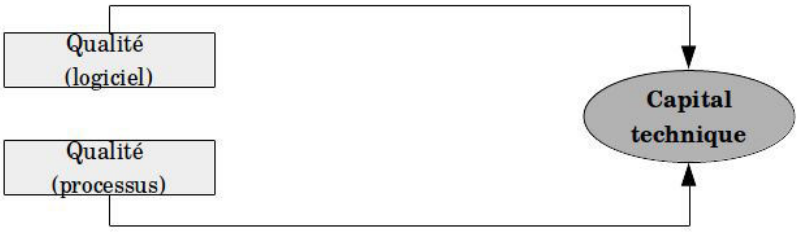


Figure 1. Notation du capital technique.

La qualité logicielle est une discipline bien établie, permettant de s'appuyer sur des approches reconnues par les professionnels. En particulier, la qualité logicielle fait l'objet de normes offrant déjà une décomposition analytique en sous-critères. La norme ISO 9126 (www.iso.org) propose ainsi 6 critères de qualité (fonctionnalité, fiabilité, utilisabilité, efficience, maintenabilité et portabilité), eux-mêmes décomposés en sous-critères. L'analyse peut porter sur le logiciel en lui-même, c'est-à-dire son code source, ou sur le processus de développement.

La qualité du logiciel (code source) peut faire l'objet d'une évaluation automatique sur base de métriques, calculées à l'aide de logiciels spécialisés. Les plus avancés permettent le calcul d'indices de maintenabilité (notation) ainsi que d'une dette technique. En pratique, la dette technique peut être calculée comme l'effort requis pour corriger les défauts (p. ex. complexité inutile, manque de commentaires ou non respect des conventions de codage) qui subsistent dans le code source. Dans le cas du logiciel *open source* SonarQube (www.sonarqube.org), le calcul de la dette technique se fait sur base d'une analyse statique du code source et s'appuie sur le modèle SQALE (www.sqale.org). Déduit de la dette technique, le ratio de dette technique représente le rapport entre la dette technique et l'effort de développement (calculable par exemple avec la méthode COCOMO).

Capital communautaire

Le capital communautaire peut être noté à partir de l'activité sur le logiciel, liée à la taille et la diversité de la communauté, de la gouvernance du projet (favorable ou non à la coopération ouverte) et du processus d'animation de la communauté (Figure 2).

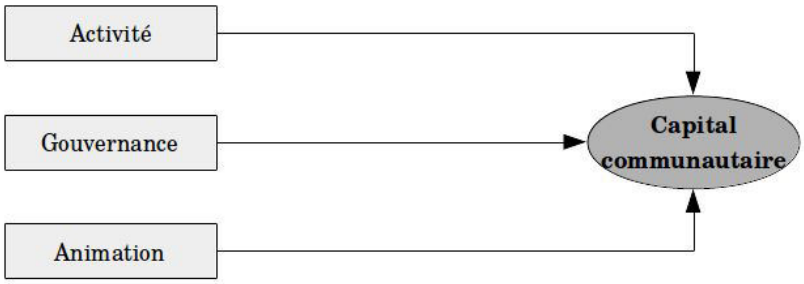


Figure 2. Notation du capital communautaire.

La santé de la communauté se traduit notamment par l'activité de ses membres. Cette activité se décline en activités au niveau du développement (p. ex. *commits*), des *mailing lists* et du *bug tracker* (Goeminne & Mens, 2011). La mesure de l'activité peut, jusqu'à un certain point, faire l'objet d'une évaluation automatique. Si la littérature pionnière sur le modèle de développement *open source*, et en particulier les écrits d'Eric Raymond (1999), peuvent donner une impression de spontanéité, l'activité est fortement liée au choix de gestion en termes de gouvernance du projet et d'animation. Markus (2007) définit la gouvernance *open source* comme l'ensemble des moyens mis en œuvre pour l'orientation, le contrôle et la coordination d'organisations et d'individus totalement ou partiellement autonomes pour le compte d'un projet de développement *open source* auquel ils contribuent collectivement. L'ouverture d'un projet limite le contrôle individuel sur le projet mais contribue par contre à la diversité des acteurs contribuant sur le projet, comme semble notamment le montrer LibreOffice.org, le *fork* d'OpenOffice.org (Gamalielsson & Lundell, 2012). L'évaluation de la gouvernance est possible grâce à l'Open Governance Index mis au point et documenté par Lizz Laffan (2011, 2012). L'activité peut aussi être soutenue par un travail d'animation spécifique (Viseur, 2007), soutenu par un *community manager* au faite des spécificités de l'*open source* et des attentes des contributeurs vis-à-vis de la fonction (Mäenpää et al., 2017).

Capital marque

Avec l'investissement des entreprises dans le secteur *open source*, la marque est devenue un levier essentiel dans les stratégies commerciales *open source* (Fitzgerald, 2006).

Le capital marque peut être noté à partir du processus de promotion, de la notoriété, elle-même fonction du nombre de clients et du nombre d'utilisateurs, et de la réputation du projet (Figure 3).

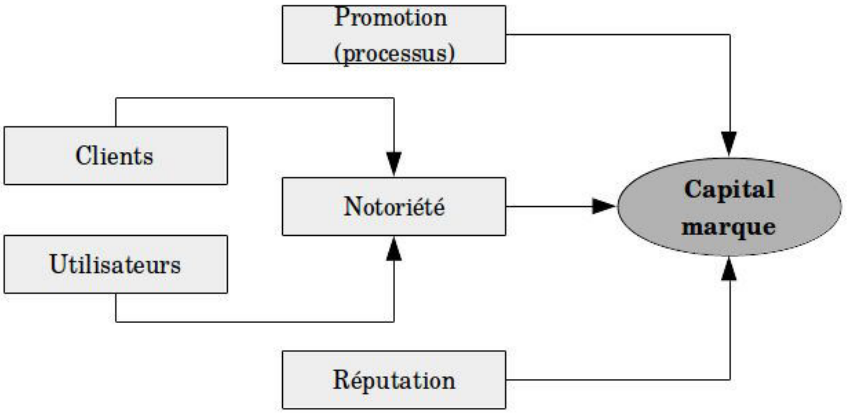


Figure 3. Notation du capital marque.

Le processus de promotion s'intéresse à ce qui est mis en place pour faire connaître le projet, pour diffuser de l'information, etc. La mesure du nombre de clients et du nombre d'utilisateurs, conditionnant la notoriété du projet, peut s'appuyer sur les mesures de comparaison (p. ex. part de voix[5] et Google Trends). La réputation du projet nécessite pour sa part la mise en place d'outils et de méthodologies permettant de réduire la subjectivité (p. ex. échelle et analyse de sentiments).

Capital client

Le capital client évalué est celui de l'écosystème du projet. En effet, un projet *open source* n'est pas nécessairement accolé à une entreprise. Il est évalué suivant trois dimensions : la dynamique d'évolution de la technologie, la dynamique d'évolution des besoins et la solvabilité de la demande. Certains utilisateurs peuvent avoir besoin de services complémentaires pour utiliser le logiciel. Comme indiqué précédemment, cela dépendra des caractéristiques du logiciel, de sa dynamique technologique et de sa couverture fonctionnelle, et de la solvabilité de ces besoins.

[5] La part de voix est ici définie par analogie à la part sociale de voix (*social share of voice*). Cette dernière fait référence à l'importance d'un terme comparé à un ensemble de termes dans les médias sociaux (Emerson et al., 2012).

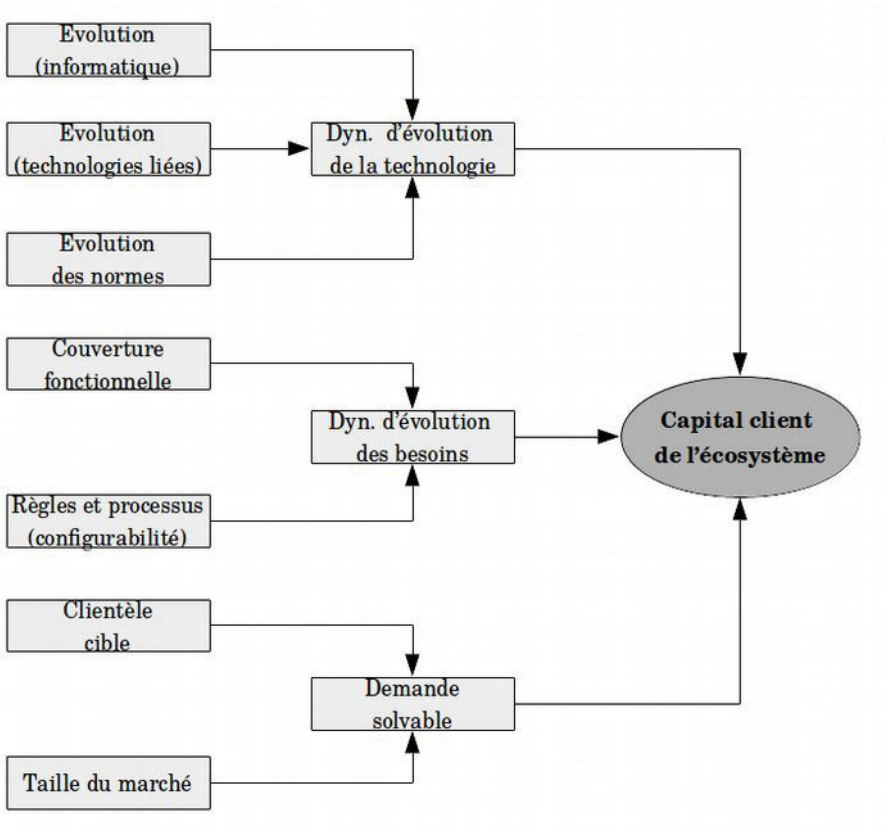


Figure 4. Notation du capital client.

La dynamique d'évolution de la technologie couvre l'évolution informatique (p. ex. *cloud computing*), l'évolution des technologies liées (p. ex. bases de données) et l'évolution des normes (p. ex. RGPD). Une dynamique forte d'évolution de la technologie entraîne la nécessité de capacités techniques entretenues et justifie par exemple la présence d'un éditeur.

La dynamique d'évolution des besoins repose sur la couverture fonctionnelle ainsi que sur la configurabilité des règles et des processus. Une forte dynamique d'évolution des besoins suppose des capacités humaines (p. ex. paramétrage et développement de modules complémentaires) voire de capacités techniques entretenues (p. ex. portages en informatique embarquée).

La demande solvable repose sur la clientèle cible et sur la taille du marché. La taille moyenne des clients cibles va conditionner la propension à payer. Combinée à la taille (en termes de nombre de clients), elle détermine le potentiel lié à la demande. Les professionnels individuels et les TPE (Très Petites Entreprises) sont nombreux mais recourent davantage au piratage ou aux solutions SaaS plus économiques. Les PME (Petites et Moyennes Entreprises) peuvent bénéficier de solutions de type SaaS ou souhaiter disposer de solutions *on-premises* entraînant le recours à une assistance de bas niveau et/ou l'achat de *packages* complémentaires. Leurs capacités financières limitent en effet souvent les possibilités d'investissement

dans des solutions informatiques davantage personnalisées. Les ETI (Entreprises de Taille Intermédiaires) et les grands comptes peuvent commander de l'assistance ou de l'adaptation afin de satisfaire des besoins plus spécifiques (p. ex. champions cachés). Ils peuvent aussi avoir besoin de TMA (Tierce Maintenance Applicative), voire d'offres SaaS si le logiciel n'est pas critique (p. ex. risque de *lock-in*).

5.2. Estimation de la valeur

Valeur de production d'un logiciel

Boehm (1981) a développé une série de trois modèles sous l'appellation générique COCOMO (Constructive COst MOdel). Ces modèles sont baptisés Basic COCOMO, Intermediate COCOMO et Detailed COCOMO (Boehm, 1981). Basic COCOMO est conçu pour fournir des estimations rapides, précoces et approximatives dans le cadre d'une première évaluation du coût de développement (Kitchenham & Taylor, 1984). Il est ensuite adapté en fonction du type de logiciel étudié (Basic COCOMO Organic / Semi-detached / Embedded).

Sur un plan pratique, la méthode Basic COCOMO fournit un ensemble de coefficients *a*, *b*, et *c* permettant, sur base du nombre de milliers de lignes de code source (KDSI), de calculer le nombre d'homme.mois nécessaires (MM) ainsi que la durée du développement logiciel (TDEV). Sur cette base, la connaissance du salaire chargé pour un développeur informatique permet de calculer une estimation de la valeur de production du logiciel.

```
MM    = a . KDSI b
TDEV  = 2,5 . MM c
```

La plate-forme OpenHub (www.openhub.net) utilise par exemple Basic COCOMO (Organic) pour l'estimation de la valeur de logiciels *open source* dont le code a été rendu public. Une approche similaire a également été développée par McPherson et al. (2008) pour estimer la valeur du noyau Linux et d'une distribution GNU/Linux.

Le modèle COCOMO, bien étudié, connu par les professionnels et couramment réutilisé pour l'estimation de la valeur de production d'un logiciel, sera dès lors utilisé pour estimer la valeur du code source initial ainsi que des améliorations amenées au fil du temps sur le logiciel *open source*.

Capital immatériel (projet open source)

La méthode COCOMO permet, à partir de la connaissance du nombre de milliers de lignes de code, de calculer une estimation du coût de développement. De manière à refléter la valeur réelle du développement, les coûts de non qualité peuvent être retranchés à l'aide du ratio d'efficience. Le ratio d'efficience traduit en pratique la capacité de l'équipe de développement à produire du code avec un niveau de qualité attendu (Devos et al., 2013). Il peut être défini comme le ratio entre l'effort de développement réduit par la dette technique et l'effort de développement.

```
Capital technique =
      coût de développement estimé
x     efficience
```

La même approche peut être appliquée pour le calcul de la valeur des contributions apportées, tous contributeurs confondus, sur le projet, en intégrant en

plus un taux de croissance prévisionnel, sur la période de retour sur investissement envisagée (*payback period*).

```
Capital communautaire =
      coût de développement estimé
x     efficience
x     (1 + taux de croissance)
x     temps de retour
```

Un état de l'art sur les méthodes d'évaluation des marques est fourni par Salinas et Ambler (2009). Trois grandes approches sont identifiées par les auteurs. La première consiste à calculer la valeur de création de la marque sur base des investissements antérieurs. La seconde consiste à comparer la marque à des transactions antérieures. La troisième consiste à évaluer la marque sur base des revenus qu'elle génère. Pour une marque propriété d'un éditeur, l'approche la plus adaptée paraît l'estimation de la valeur de re-création fondée sur une évaluation de l'historique des dépenses, éventuellement basée sur la disponibilité des comptes annuels de l'entreprise. En l'absence d'entreprise, l'estimation pourrait se faire par comparaison à des projets équivalents déjà évalués (p. ex. classements Interbrand ou BrandZ).

Capital immatériel contrôlé

L'entreprise ne possède pas en propre le projet *open source*. L'entreprise dispose d'un certain degré de contrôle du code source du logiciel et est capable de capter une partie de la notoriété associé à son nom.

Levier essentiel dans les stratégies commerciales *open source*, la marque est également un élément clef dans la gouvernance du projet. En particulier, elle peut être propriété soit de l'entreprise soit d'une fondation créée pour soutenir le projet (Riehle, 2010). La propriété de la marque associée au projet permet à l'entreprise de capter la totalité de la notoriété associée au projet et constitue un actif valorisable.

```
Valeur du capital immatériel contrôlé =
      capital technique
x     degré de contrôle
+     capital communautaire
x     degré de contrôle
+     capital marque
x     degré de captation
```

Le degré de contrôle peut être estimé à :

50 % si accord de contributeur
50 % si licence permissive (appropriabilité)
part des lignes de code
si licence *copyleft* et pas d'accord de contributeur
ou si part des lignes de code > 50 %

Conventionnellement, le degré de contrôle peut être pris à 80 % en cas d'éditeur *open source* et en l'absence d'information plus précise (Capra et al., 2008).

Le degré de captation peut être estimé à :

0 %	si pas de propriété de la marque
100 %	si propriété de la marque

ou

part de voix[6]	si pas de propriété de la marque

5.3. Exemple (estimation de la valeur)

Le logiciel `<fictivesoftware>` comptabilise environ 305k lignes de code source[7] et représente un effort estimé de 80 développeurs.an (Basic COCOMO). Sa valeur est estimée à 4 millions d'euros. Le ratio de dette technique est de 4,5%, si bien que l'efficience est de 95,5%. L'entreprise produit un quart du code source de l'application et impose un accord de contributeur aux développeurs, si bien que le degré de contrôle est de 50%. La communauté produit 30k lignes de code par an; sa valeur est estimée à 400k euros. L'Open Governance Index est de 40% et devrait rester stable ; toutes autres choses restant égales, le taux de croissance annuel devrait être nul (0,0%). Le nom du logiciel est protégé comme marque par l'entreprise. Sa valeur est estimée à 1 million d'euros sur base de sa valeur de recréation. Le *cash flow* cumulé actualisé, calculé sur base des bilans de l'entreprise, s'élève à 5 millions d'euros sur 3 ans.

```
Valeur du capital immatériel (hors capital client)
=     € 4.000.000
x     95,5%
+     € 400.000
x     3
x     (1 + 0,0%)
=     € 5.020.000

Valeur du capital immatériel contrôlé
=     € 5.020.000
x     50,0%
+     € 1,000,000
x     100.0%
=     € 3.510.000
```

6. Discussion

Nos résultats préliminaires débouchent sur une architecture d'évaluation d'un logiciel *open source* incluant sa notation ainsi que l'estimation de sa valeur compte tenu de son capital technique, de son capital communautaire et de son capital marque. Outre l'exemple fictif utilisé comme illustration, l'approche a fait l'objet d'un premier test (en cours) sur un projet concret (ERP open source Odoo) retenu pour sa popularité et la disponibilité d'informations tant sur le développement que sur les finances de l'entreprise associée, permettant le calcul du *cash flow* cumulé

[6] Nous proposons ici de mesurer le degré de captation via la part de voix en estimant, par exemple à l'aide d'un moteur de recherche, donc parmi l'ensemble des documents indexés, le taux d'association entre le nom de l'entreprise exploitant le logiciel *open source* et le nom du logiciel *open source*.

[7] Cette information peut être obtenue soit par la consultation de sites publics (p. ex. Openhub) soit par l'utilisation de logiciels *open source* spécialisés comme Ohcount (comptage de lignes de code source) ou Sonarqube (évaluation de la qualité du code source).

actualisé et une estimation basse de la valeur de la marque sur base des budgets marketing estimés.

L'approche présente plusieurs limitations. Premièrement, la méthodologie de notation propose des critères et des sous-critères mais n'est pas encore accompagnée d'une grille d'analyse avec une échelle de mesure par sous-critères permettant de réduire la subjectivité de l'évaluateur. Cette amélioration pourra cependant s'appuyer sur les travaux antérieurs, qu'il s'agisse de la méthode QSOS (2013) ou de la méthode OGI (Laffan, 2011, 2012) proposant déjà des grilles documentées, publiées et réutilisables. Deuxièmement, l'approche proposée constitue une ébauche et nécessite d'être validée sur un ensemble d'études de cas. Quatre cas sont actuellement envisagés : Gimp (projet de type « engagement »), Linux (projet de type « exploration »), Odoo (en cours) et Adacore (projets entre « optimisation » et « expertise »). L'extension à l'écosystème Claroline (projets Claroline Connect et Chamilo) est également possible, sur base des travaux actuels de Viseur et al. (2018). À plus long terme, le modèle de notation doit soutenir la mise en place d'un outil d'aide à la décision permettant, compte tenu de la qualité des actifs et des objectifs de développement, de calibrer le montant et la ventilation des investissements réalisés sur le projet.

7. Références

Boehm, B.W. (1981). « Software engineering economics », Prentice-Hall.

Bonnet, P. (2011). « Mesure de la valeur des actifs immatériels du système d'information - Valeur intrinsèque des données, règles et processus », Hermès - Lavoisier.

Capra, E, Francalanci, C, Merlo, F. (2008). « An empirical study on the relationship between software design quality, development effort and governance in open source projects », IEEE Transactions on Software Engineering, 34(6), 765-782.

Charleux, A., Mione, A. (2018). « Les business models de l'édition open source ; Le cas des logiciels », Finance Contrôle Stratégie.

Chesbrough, H., Vanhaverbeke, W., West, J. (2006). « Open Innovation Researching a New Paradigm », Oxford, Oxford University Press.

Dahlander, L., & Wallin, M. W. (2006). « A man on the inside: Unlocking communities as complementary assets ». *Research Policy*, *35*(8), 1243-1259.

Deprez, J. C., & Alexandre, S. (2008). « Comparing assessment methodologies for free/open source software: OpenBRR and QSOS. In International Conference on Product Focused Software Process Improvement », Springer, Berlin, pp. 189-203.

Devos, N., Durieux, D., Ponsard, C. (2013). « Managing technical debt in IT start-ups – an industrial survey, International Conference on Software and System Engineering and their Applications (ICSSEA) », November 2013, Paris (France).

Emerson, T., Ghosh, R., Smith, E. (2012). « CASE STUDY: Using the Social Share of Voice to Predict Events That Are about to Happen », Practical Text Mining and Statistical Analysis for Non-Structured Text Data Applications, pp. 127-131.

Fitzgerald, B. (2006). « The transformation of open source software ». Mis Quarterly, pp. 587-598.

Fraix, J. (1988). « Manuel d'évaluation des projets industriels », De Boeck Université (ISBN: 978-2804111342).

Fustec, A, Marois, B. (2006). « Valoriser le capital immatériel de l'entreprise », Éditions d'organisation. ISBN : 978-2708136656.

Fustec, A. (2016). « Évaluation du capital intellectuel par des indices de notation, profitabilité et performances financières des entreprises », Innovations, 51,(3), 125-146. doi:10.3917/inno.051.0125.

Gadrey, J. (2003). « Socio-économie des services », La Découverte, Paris.

Gamalielsson, J., & Lundell, B. (2012). « Long-term sustainability of open source software communities beyond a fork: A case study of libreoffice ». In IFIP International Conference on Open Source Systems. Springer, Berlin, pp. 29-47.

Goeminne, M., & Mens, T. (2011). « Evidence for the pareto principle in open source software activity ». In the Joint Proceedings of the 1st International workshop on Model Driven Software Maintenance and 5th International Workshop on Software Quality and Maintainability, pp. 74-82.

Izquierdo-Cortazar, D., Gonzalez-Barahona, J. M., Duenas, S., & Robles, G. (2010). « Towards automated quality models for software development communities: The QualOSS and FLOSSMetrics case ». In Seventh international conference on the quality of information and communications technology (Quatic), IEEE , pp. 364-369.

Jullien, N., Zimmermann, J. B. (2009). « Firms' contribution to open□source software and the dominant user's skill ». European Management Review, 6(2), 130-139.

Jullien, N., Zimmermann, J.-B. (2011). « Floss firms, users and communities: a viable match? », Journal of Innovation Economics & Management, 2011/1 (n°7), p. 31-53.

Jullien, N. & Zimmermann, J.B. (2013). « Le logiciel libre : un renouveau du modèle industriel coopératif de l'informatique ». In Paloque-Berges C. & Masutti C. (Éds.), Histoires et cultures du Libre. Des logiciels partagés aux licences échangées, Framabook.

Kitchenham, B.A., Taylor N.R. (1984). « Software cost models », ICL technical journal.

Kogut, B., & Metiu, A. (2001). « Open□source software development and distributed innovation ». Oxford review of economic policy, 17(2), 248-264.

Laffan, L. (2011). « Open governance index-Measuring the true openness of open source projects from Android to WebKit », VisionMobile, London. Online: https://upload.wikimedia.org/wikipedia/commons/5/5f/VisionMobile_Open_Governance_Index_report.

Laffan, L. (2012). « A new way of measuring openness: The open governance index », Technology Innovation Management Review, vol. 2, n° 1. Online: http://www.timreview.ca/article/.

Mankins, J.C. (2009). « Technology readiness and risk assessments: A new approach », Acta Astronautica, 65(9-10), pp. 1208-1215.

Markus, M. L. (2007). « The Governance of Free/Open Source Software Projects: Monolithic, Multidimensional, or Configurational? », Journal of Management & Governance, 11(2), pp. 151-163.

McPherson, A., Proffitt, B, HaleEvans, R. (2008). Estimating the total development cost of a Linux distribution. The Linux Foundation, October 2008.

Miloud, T. & Cabrol, M. (2011). « Les facteurs stratégiques influençant l'évaluation des start-ups par les capitaux-risqueurs », Management & Avenir, 49,(9), 36-61. doi:10.3917/mav.049.0036.

Muselli, L. (2008). « Le rôle de licences dans les modèles économiques des éditeurs de logiciels open source », Revue française de gestion, n° 181, pp. 199-214.

Pénin, J. (2011). « Open source innovation: Towards a generalization of the open source model beyond software ». Revue d'économie industrielle, n° 4, pp. 65-88. Online: http://journals.openedition.org/rei/5184.

QSOS (2013). « Qualification et Sélection de logiciels Open Source (QSOS) Version 2.0 - 19/01/2013 »,

Riehle, D. (2010). « The economic case for open source foundations ». Computer, 43(1), pp. 86-90.

Raymond, E. (1999). « The Cathedral and the Bazaar, Knowledge », Technology and Policy, 12(3), pp. 23-49.

Salinas, G., Ambler, T. (2009). « A taxonomy of brand valuation practice: Methodologies and purposes », Journal of Brand Management, 17(1), pp. 39-61.

Schaarschmidt, M., Walsh, G., Von Kortzfleisch, H. F. O. (2015). How Do Firms Influence Open Source Software Communities? A Framework and Empirical Analysis of Different Governance Modes, Information and Organization, 25(2), 99-114.

Viseur, R. (2007). « Gestion de communautés Open Source », 12ème Conférence de l'Association Information et Management, Lausanne (Suisse).

Viseur, R. (2013). « Évolution des stratégies et modèles d'affaires des éditeurs open source face au cloud computing », Terminal, 113-114, 173-193.

Viseur, R. (2018). « Gouvernance des projets open source : le cas du logiciel Claroline ». In Congrès INFORSID, Nantes (France).

Von Hippel, E. (2001). « Innovation by user communities: Learning from open-source software ». *MIT Sloan management review*, *42*(4), 82-82.

West, J. (2003). « How open is open enough?: Melding proprietary and open source platform strategies ». Research policy, 32(7), pp. 1259-1285.

Alerter ou ne pas alerter ? Telle est la question...

Maude Arru, Elsa Negre, Camille Rosenthal-Sabroux

Paris-Dauphine University, PSL Research Universities, CNRS UMR 7243, LAMSADE, 75016 Paris, France

maude.arru@dauphine.fr

RÉSUMÉ. *La plupart des crises, environnementales, humanitaires, économiques ou même sociales, surviennent après différents signaux avant-coureurs permettant de déclencher des alertes. Ces alertes peuvent aider à prévenir des dommages si elles sont lancées en temps voulu ainsi qu'à fournir des informations pour permettre aux intervenants et à la population à se préparer de manière adéquate à la crise à venir. Aujourd'hui, de nombreux systèmes basés sur les technologies de l'information et de la communication sont conçus pour reconnaître les signaux imminents de crise afin d'en limiter les conséquences. Les systèmes d'alerte en font partie, ils se sont révélés efficaces, mais comme pour tous les systèmes incluant des êtres humains, une part d'imprévu subsiste. Nous proposons donc une méthode d'analyse de données qui permet aux décideurs des cellules de crise de disposer d'éléments de réponse à la question d'alerter ou non les populations dans une zone géographique donnée. Cette méthode est basée sur une sélection de facteurs influençant les comportements de la population, pour lesquels nous établissons une liste d'indicateurs pertinents pouvant être renseignés avant ou au cours de la phase préliminaire d'une crise dans les systèmes d'alerte. À partir de ces indicateurs, nous proposons un outil d'aide à la décision (basé sur un arbre de décision en tant que représentation possible).*

ABSTRACT. *Most of crises, environmental, humanitarian, economic or even social, occur after different presaging signals that permit to trigger warnings. These warnings can help to prevent damages and harm if they are issued timely and provide information that enables responders and population to adequately prepare for the disaster to come. Today, there are many systems based on Information and Communication Technologies that are designed to recognize foreboding signals of crises to limit their consequences. Warning system are part of them, they have proved to be effective, but as for all systems including human beings, a part of unpredictable remains. In this article, we provide a method of data analysis that allows decision makers in crisis cells to have answer elements to the question of alerting or not populations in a given geographical area. This method is based on a selection of factors that influence population behaviors, for which we establish a list of relevant indicators that can be informed before or in the preliminary phase of a crisis into warning systems. From these indicators, we propose a tool for decision support (based on a decision tree as a possible representation).*

MOTS-CLÉS : *Gestion de crise, Systèmes d'alertes (précoces), Aide à la décision*

KEYWORDS: *Crisis Management, (Early) Warnig Systems, Decision Support*

L'insertion dans les systèmes d'information d'éléments cognitifs et de simulation de comportements humains réalistes pour reproduire ou prédire des événements ou des actions constitue un défi pour les développeurs car cela nécessite l'interconnexion d'éléments hétérogènes pouvant être physiologiques, psychologiques, sociaux ou environnementaux. Aujourd'hui, grâce aux progrès de la gestion de données, il est plus rapide et efficace de travailler en temps réel, créer des cartes à partir de données géolocalisées ou réaliser des évaluations sur la base de scénarios intégrant des données de différentes sources. Ces évolutions ont permis d'améliorer les systèmes de gestion de crise, systèmes d'information développés pour aider les acteurs amenés à prendre des décisions lors d'une catastrophe. En effet, des décisions importantes doivent être prises avant, pendant et après une crise. Elles reposent sur des données et informations objectives, mais sont également déterminées par des éléments subjectifs tels que des biais cognitifs pouvant limiter l'efficacité de la réponse. Une manière de réduire l'effet des biais cognitifs est d'améliorer l'exhaustivité de l'information dans les systèmes de gestion de crise. Ces derniers sont des outils qui permettent notamment de prévoir de manière aussi précise et rapide que possible les conséquences d'une crise et son évolution sur un territoire donné tout en prenant en compte des informations de plus en plus complexes. En effet, les systèmes de gestion de crise intègrent des données de différentes sources et natures. Cependant, malgré les connaissances et les technologies développées pour minimiser ou éviter les conséquences désastreuses qu'une crise peut avoir, des crises demeurent, en partie, déterminées par des phénomènes incertains, qui ne sont pas toujours pris en compte, comme par exemple, la vulnérabilité des territoires, le besoin de coordination entre les services et les comportements probables des populations en danger (Arru *et al*., 2018).

L'article résumé ici (Arru *et al*., 2019) porte sur les systèmes d'alerte précoce. Plusieurs acteurs gravitent autour de ces systèmes d'alerte avec des rôles différents. Les principaux acteurs sont les spécialistes de la gestion de crise et les experts qui construisent des modèles et contribuent à alimenter le système d'alerte, les décideurs qui agissent pour la résolution des crises, les acteurs du terrain qui appliquent les décisions prises dans la cellule de crise et enfin les populations. Cet article porte sur deux de ces catégories d'acteurs, les décideurs et les populations, en proposant une analyse basée sur les comportements de ces derniers en situation d'alerte et de crise. La prise en compte des lois et phénomènes régissant les comportements en situation de crise nous semble un axe important de recherche et de réflexion sur l'amélioration de la diffusion de l'alerte, la communication de crise et le développement de politiques d'éducation et de sensibilisation ciblées. En effet, avant et pendant une crise, les individus agissent en fonction de leurs propres connaissances et schémas d'interprétation. Ces schémas ne permettent pas toujours aux personnes de réagir de manière appropriée aux situations à risque et peuvent entraîner des réactions dangereuses (Mileti, Sorensen, 1990). De plus, de nombreuses recommandations préconisent de recentrer les systèmes d'alerte sur l'être humain, principalement par la participation des populations au processus de prise de décision (Basher, 2006). Il nous semble complémentaire

à cette approche d'intégrer ces aspects centrés sur l'humain dans la connaissance du risque et dans la sensibilisation faite à travers la connaissance de ses comportements.

Avertir peut aider à faire face à une crise en protégeant les populations, mais cela peut aussi constituer une menace et avoir des effets plus néfastes que ceux de la crise. Ainsi, afin d'améliorer l'adaptation des systèmes d'alerte aux populations concernées, nous proposons dans (Arru *et al*., 2019) une méthode pour aider les décideurs (souvent en cellule de crise) à déterminer s'ils doivent ou non alerter les populations en fonction de leurs comportements probables. Nous commençons par définir les principaux concepts liés à notre proposition puis nous présentons notre processus d'aide à la décision. Ce processus a pour objectif de fournir un modèle qui permet d'aider à déterminer si les populations doivent être alertées en fonction de leurs comportements probables basés sur des facteurs/indicateurs reconnus pour avoir un impact sur les comportements. Nous précisons que ces indicateurs ne sont pas des signaux faibles d'une potentielle crise, ils ont été identifiés a posteriori pour leur influence sur les comportements. Afin de valider la faisabilité de notre approche, nous l'avons appliquée à des données réelles relatives à neuf cas d'accidents technologiques survenus en France entre 1981 et 2013. Pour chaque accident nous avons collecté des informations correspondant à 14 indicateurs ayant un impact sur les comportements des populations. Enfin, à partir de ces données, nous avons généré un arbre de décision (à partir de l'algorithme Random tree (Kalmegh, 2015)) destiné aux décideurs d'une cellule de crise et ainsi les aider à déterminer de la pertinence d'une alerte aux populations.
Dans nos travaux futurs, cette approche devra être validée par une analyse croisée entre experts en risques de différents domaines. Il sera nécessaire d'identifier les caractéristiques précises de l'alerte et de la réponse en fonction de la typologie des crises afin que les différents facteurs/indicateurs et décisions puissent être sélectionnés de manière appropriée. Enfin, il faudra veiller à récupérer des données provenant de différentes sources dans un outil pouvant être intégré à une cellule de crise.

Bibliographie

Arru M., Negre E., Rosenthal-Sabroux C. (2018). Population behaviors in crisis situations - A study of behavioral factors in the PPI ineos emergency response exercise. In *51st hawaii int. conf. on system sciences, HICSS 2018, hawaii, usa, january 3-6, 2018.*

Arru M., Negre E., Rosenthal-Sabroux C. (2019). To alert or not to alert? that is the question. In *52nd hawaii int. conf. on system sciences, HICSS 2019, hawaii, usa, january 8-11, 2019.*

Basher R. (2006, 09). Global early warning systems for natural hazards: Systematic and people-centred. *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences*, vol. 364, p. 2167-82.

Kalmegh R. (2015). Comparative analysis of weka data mining algorithm randomforest , randomtree and ladtree for classification of indigenous news data sushilkumar. *International Journal of Emerging Technology and Advanced Engineering*, vol. 1, n° 5.

Mileti D., Sorensen J. (1990, 8). Communication of emergency public warnings: A social science perspective and state-of-the-art assessment. , n° ORNL-6609.

Le Rôle des Ressources dans l'Evolution des Systèmes d'Information

Manuele Kirsch Pinheiro[1], Carine Souveyet[1]

1. Centre de Recherche en Informatique, Université Paris 1 Panthéon Sorbonne
90 rue de Tolbiac, 75013 Paris, France
Manuele.Kirsch-Pinheiro@univ-paris1.fr, Carine.Souveyet@univ-paris1.fr

RESUME. *L'introduction de nouvelles pratiques, comme le Cloud Computing, l'IoT ou le Fog Computing, transforme les SI et nous oblige à mieux considérer la position des ressources dans ces systèmes. Celles-ci sont devenues hétérogènes, dynamiques et assument un rôle de plus en plus stratégique dans les SI. Nous discutons ici le rôle des ressources dans l'évolution des SI vers des SI dits Pervasifs et proposons leur prise en charge lors de la conceptualisation de ces systèmes à travers une extension du cadre conceptuel de l'espace de services.*

ABSTRACT. *On this paper, we propose to extend the conceptual framework named Space of Service in order to consider the resources usage on the Information System (IS) evolution. Resources represent now a strategic aspect of IS, thanks to new practices such as Cloud Computing, IoT or Fog Computing. Such practices transform IS resources, that become more dynamic, heterogeneous, and whose usage becomes more strategic for organizations. Including resources on a conceptual analysis of IS turns into a necessity for following the evolution of IS into new Pervasive IS.*

Mots-clés : Systèmes d'Information Pervasifs, gestion de ressources, services.

KEYWORDS: *Pervasive Information Systems, resource management, services.*

1. Introduction

Les Systèmes d'Information (SI) sont en évolution. L'intégration de nouvelles technologies (*IoT*, *Cloud & Fog Computing*, micro-services, etc.) et des nouvelles pratiques (BYOD[1], consommation à la demande, etc.) a ouvert des nouvelles perspectives pour ces systèmes. Ceux-ci se retrouvent libérés des frontières de l'organisation. Ils s'externalisent, s'étendent désormais sur l'environnement physique, accompagnent leurs acteurs où qu'ils soient. On assiste à une transformation majeure dans laquelle les SI d'aujourd'hui deviennent petit à petit des SI Pervasifs (SIP). A la différence des SI traditionnels, les SIP visent un environnement dynamique et hétérogène, composé d'une multitude d'artefacts capables de percevoir le contexte

[1] *Bring Your Own Device*, pratique consistant à utiliser son matériel personnel (ordinateur portable, tablette, smartphone…) en entreprise.

de l'utilisateur et de gérer sa mobilité ; ils s'intègrent progressivement à l'environnement physique et prônent une interaction continue avec l'utilisateur, afin de mieux répondre à leurs besoins (Kourouthanassis et Giaglis, 2006).

Cette nouvelle génération des SI manque encore d'outils conceptuels permettant de mieux maitriser sa complexité, ouvrant la voie à une meilleure gestion dans sa globalité. Des initiatives existent pourtant (Rao et Angelov, 2009 ; Kourouthanassis *et al.*, 2010), dont la notion d'espace de services (Kirsch Pinheiro *et al.*, 2013). Celle-ci se présente comme un cadre conceptuel permettant d'analyser un SIP tout en faisant abstraction de l'hétérogénéité inhérente à ces systèmes. Elle fonde son analyse sur deux abstractions majeures : les services (les services offerts par le système) et le contexte (les éléments de contexte observables dans l'environnement). Or un troisième élément manque dans cette analyse : les ressources disponibles.

L'introduction des ressources de plus en plus hétérogènes et dynamiques, comme le *Cloud Computing* (Mell et Grance, 2011) et l'*IoT* (Gubbi *et al.* 2013), permet aujourd'hui d'envisager un usage opportuniste des ressources disponibles, tel que soutenu par le *Fog Computing* (Villari et al. 2016) ou les grilles pervasives (Parashar et Pierson, 2010). La gestion de ressources devient ainsi un axe de réflexion stratégique pour les SI. La conceptualisation des nouveaux SI Pervasifs doit donc prendre en considération ce nouvel axe au même titre que les services proposés par le système. Nous proposons ainsi d'étendre le cadre conceptuel des espaces de services (Kirsch Pinheiro *et al.*, 2013) afin de tenir compte de la disponibilité des ressources pour l'exécution des services proposés dans le cadre d'un SIP.

Cet article s'organise comme suit : dans la section 2, nous discutons la gestion des ressources sur les SI ; dans la section 3, nous présentons notre extension de l'espace de service et nous l'illustrons par un exemple dans la session 4 ; avant de conclure dans la section 5.

2. La gestion de ressources dans les SI

Pendant des années, la notion de ressources permettant l'exécution de services n'a reçu que peu d'attention dans la modélisation et la conceptualisation des SI. Ceci s'explique en grande partie par le fait que les ressources disponibles dans un SI étaient majoritairement stables et homogènes. On parle notamment des « *data centers* » et de structures semblables, où tous les calculs (c.a.d. l'exécution des services proposés par le SI) ont lieu. Les ressources n'étaient ainsi pas perçues comme quelque chose de stratégique pour le SI : quel que soit le service, il serait exécuté dans ces structures avec une infrastructure plutôt stable. Or l'introduction du *Cloud Computing*, et plus récemment du *Fog Computing*, change la perception de ces ressources.

Le modèle économique associé au *Cloud Computing* (Mell et Grance, 2011) permet la mise en place d'une analyse stratégique sur l'usage des ressources internes versus la « location » des ressources sur le *cloud*. Contrairement aux ressources internes à l'organisation, les ressources sur le *cloud* sont perçues comme ayant un faible coût de maintenance, adoptant un modèle à la demande où on peut adapter sa consommation en fonction de ses besoins et ne payer que les ressources effectivement

consommées. Cependant, l'adoption du *cloud* est également accompagnée de quelques craintes liées à l'externalisation des données et de leur traitement. Ces craintes concernent notamment la sécurité, la confidentialité ou encore la latence réseau. La dépendance envers un fournisseur extérieur peut également inquiéter pour certains services de l'organisation, tout comme les coûts de ce nouveau modèle de consommation à long terme. Le choix entre l'usage des ressources locales à l'organisation ou l'adoption d'un modèle *cloud* pour l'exécution de certains services devient alors stratégique, en plus de technique.

Parallèlement, l'introduction croissante de ressources IoT pose la question du traitement des données issues de ces dispositifs. Certains travaux suggèrent l'usage des plateformes *cloud* pour ce traitement (Gubbi *et al.,* 2013 ; Mulfari *et al.,* 2015), alors que d'autres suggèrent l'usage du *Fog Computing* afin de contrer les limitations associées au *cloud* (Chen *et al.,* 2017 ; Villari *et al.*, 2016 ; Steffenel et Kirsch Pinheiro, 2015). Le *Fog Computing* peut être vu comme un nouveau paradigme permettant la distribution des calculs, du stockage et de la gestion de services au plus proche de l'utilisateur final, tout au long du continuum entre le *cloud* et les objets (IoT) et les terminaux (Chen *et al.,* 2017). D'autres concepts semblables, comme les grilles pervasives (Parashar et Pierson, 2010) ou le *Mobile Cloud* (Wang *et al.,* 2018), suggèrent l'usage de ressources environnantes (dont, par exemple, des ressources mobiles) pour une exécution opportuniste de services. Tous ces modèles permettent d'envisager l'usage des ressources autres que ceux des *data centers* ou des plateformes *cloud* pour l'exécution de services, ouvrant la perspective d'une rationalisation de l'usage des ressources disponibles.

On rajoute à cela la tendance actuelle vers un usage accru des micro-services dans les organisations, lesquels prônent un découpage plus fin des fonctionnalités permettant, par la même occasion, un déploiement plus aisé des applications (*e.g.* Mulfari *et al.*, 2015 ; Villari *et al*. 2016). Toutes les conditions sont ainsi réunies pour qu'on puisse déployer de manière plus ou moins transparente les services offerts par un SI sur des ressources aussi variées que les ressources *cloud* (privé ou public), celles des *data centers*, des dispositifs réseaux, IoT, ou encore des terminaux mobiles.

Toutes ces évolutions ont transformé la nature des ressources disponibles dans les SI. Ces ressources sont devenues davantage réparties, hétérogènes, disposées dans une infrastructure devenue elle-même plus dynamique. Le placement des services sur ces ressources devient ainsi un problème non-trivial. Ce placement peut être associé à la gestion de ressources, une question communément abordée par la communauté HPC (*High Performance Computing*), où des nombreux travaux sur l'ordonnancement de tâches ont été proposés, aussi bien sur les grappes et grilles de calculs, que sur les plateformes *cloud* (Wang *et al.,* 2018 ; Tyagiand et Gupta, 2018 ; Singh et Chana, 2016 ; Bessai, 2014 ; Maurer *et al.,* 2012). Dans les environnements pervasifs aussi, cette question a été abordée (Estublier *et al.*, 2012 ; Villari *et al.*, 2016). Par exemple, Villari *et al.* (2016) soutiennent un déploiement opportuniste d'applications, décomposées en micro-services, entre de ressources sur le « *edge* » (c.a.d. en bordure du SI, plus à proximité de l'utilisateur) et celles sur le *cloud*.

De même, des nombreuses outils et méthodes d'observation (monitoring) et de gestion de ressources sont utilisés/proposés aussi bien pour les infrastructures cloud que pour les centres de calcul (Alhamazani *et al.*, 2015 ; Da Cunha Rodrigues et al., 2016). A titre d'exemple, on peut citer Ganglia[2], pour l'observation et le suivi des ressources, ou encore Globus[3], pour la gestion de ressources sur les grilles et grappes de calcul. Ce genre d'outil (Alhamazani *et al.*, 2015 ; Da Cunha Rodrigues et al., 2016) reste bien souvent cantonné au seul niveau technique. Cette gestion ne se fait pas en coordination étroite ou en lien avec la direction stratégique au niveau SI.

Cependant, même si plusieurs travaux visent notamment les mécanismes et les algorithmes d'ordonnancement de tâches, parfois au détriment d'une vision plus systémique, ces travaux illustrent bien le challenge que représente la gestion des ressources. Celle-ci peut assumer dès à présent un rôle stratégique pour les SI dans la rationalisation de ces ressources devenues dynamiques et hétérogènes. La prise en charge des ressources disponibles dans la conceptualisation des SI devient alors un élément essentiel pour leur transformation vers les SIP.

3. L'espace de services et la gestion des ressources

La notion de services est une notion communément utilisée pour la conceptualisation de SI. Rao et Angelov (2009), par exemple, adoptent une perspective orientée service dans leur cadre conceptuel, afin de mieux centrer leur analyse sur l'offre de services et pas uniquement sur les technologies utilisées pour cette offre. Leur cadre offre une vision holistique des SIP en 4 éléments (*infostructure*, *devices*, *interfaces*, *smart spaces*) dont l'objectif serait d'analyser la chaîne de création de la valeur ajoutée dans ces systèmes. On voit apparaître dans le cadre de Rao et Angelov (2009) la notion d'infrastructure de communication (à travers l'élément *infostructure*) et celle de terminal (l'élément *device*), mais leur analyse est entièrement tournée vers la création de la valeur ajoutée et donc vers le consommateur et son expérience. C'est donc l'infrastructure et le terminal d'accès aux services par ce consommateur qui sont considérés et pas forcément ceux nécessaire à exécution du service. Leur caractère mobile et embarqué est pris en considération, mais pas la disponibilité des ressources, ni leurs capacités à offrir un certain service. Rao et Angelov (2010), tout comme Kourouthanassis *et al.* (2010), ne considèrent pas le rôle des ressources dans l'offre de services, assumant implicitement que l'exécution des services proposés est garantie, sans considérer les ressources nécessaires pour cela.

Afin de palier à ce manque d'une vision conceptuelle des ressources dans les SIP, nous proposons ici d'étendre la définition d'espace de services afin de tenir compte des ressources disponibles. L'objectif d'une telle conceptualisation est d'offrir une vision globale d'un SIP, permettant ainsi une meilleure gestion de ces systèmes et de leurs éléments, tout en faisant abstraction des technologies impliquées, gardant ainsi une certaine transparence vis-à-vis de l'hétérogénéité de l'environnement.

---

[2] http://ganglia.sourceforge.net (dernière visite : 15 avril 2019)
[3] https://www.globus.org (dernière visite : 15 avril 2019)

La notion d'espace de services (Kirsch Pinheiro *et al.,* 2013) repose sur deux abstractions majeures : les *services*, aussi nommés *entités actives*, et le *contexte*, vu à travers les *capteurs*, aussi nommés *entités passives* (car fournissant de l'information). Ces abstractions correspondent en réalités aux *rôles* joués par les éléments composant un SI, et pas forcément aux éléments eux-mêmes. En effet, différents éléments participent au bon fonctionnement d'un SI. Ces éléments peuvent avoir un rôle plus actif, offrant des services au système, mais aussi un rôle plus passif, permettant l'observation de l'environnement autour de lui, voir les deux, offrant à la fois des services et des informations sur l'environnement (environnement physique, d'exécution ou même organisationnel). La notion d'espace de services permet ainsi de conceptualiser ces éléments à travers les rôles qu'ils jouent auprès d'un SI, ce qui permet de faire plus facilement abstraction de la vraie nature de ces éléments. Ceux-ci peuvent jouer différents rôles, ils seront alors considérés sous l'angle de leurs rôles, à la fois comme une entité passive (puisqu'il est capable d'observer l'environnement) et comme une entité active (puisqu'on peut lui associer certains services). De ce fait, lorsqu'on parle d'un capteur, il ne s'agit pas forcément d'un élément permettant uniquement l'observation de l'environnement (*e.g.* un simple capteur de température intégré à l'environnement physique), il s'agit surtout d'un élément qui *peut aussi* offrir ces informations (*e.g.* un tel capteur sur un RaspberryPI). Il n'est pas nécessairement limité à cela, mais il peut aussi jouer ce rôle.

Ainsi, dans sa définition originale (Kirsch Pinheiro *et al.,* 2013), la notion d'espace de services permet une analyse sur ces deux rôles : les *services,* représenté sur l'équation (1), et le *contexte,* représenté sur l'équation (2). A travers les services, la notion d'espace de services permet de conceptualiser les fonctionnalités $\mathcal{F}$ offertes par un SI à ses utilisateurs pour satisfaire un ensemble d'intentions I. La notion de contexte permet de représenter les éléments de contexte observables dans l'environnement (physique et organisationnel), symbolisés par l'ensemble d'observations O_{cpi}. On peut également associer à un service un contexte requis ($\mathcal{CXR}$), décrivant en quelque sorte la situation idéale dans laquelle le service a plus de chances de satisfaire ses objectifs, et un contexte d'exécution ($\mathcal{CX}$), décrivant les conditions dans lesquelles le service s'exécute. On peut alors se demander : où s'exécute un service ? C'est à ce niveau qu'intervient la notion de ressources. Ce rôle, jusqu'ici ignoré, doit désormais être pris en considération.

$$sv_i = <I, \mathcal{F}, \mathcal{CX}, \mathcal{CXR}> \quad (1)$$

$$cp_i = \{ O_{cpi}, \mathcal{CX} \} \quad (2)$$

Comme on peut observer dans l'équation (1), nous avons pu généraliser la notion de service à travers ses fonctionnalités et son intention, faisant ainsi abstraction des technologies utilisées. Au-delà des aspects purement techniques, la notion de ressource peut elle aussi être généralisée. Une ressource peut représenter une ressource informatique (un serveur, un ordinateur portable, une machine virtuelle…), comme elle peut également représenter un collaborateur. Toutes les deux sont capables de réaliser un service pour le compte du SI, participant ainsi au bon fonctionnement du système dans sa totalité. Comme les rôles dans les systèmes de

workflow autorisent (ou non) l'exécution d'une tâche par un collaborateur, l'idée serait de représenter cette possibilité d'autoriser (ou non) l'exécution d'un service par une ressource, quelle que soit sa nature, en fonction de ses capacités.

En explicitant la notion de ressource dans l'espace de services, on espère pouvoir considérer le placement des services proposés par le SIP sur les ressources, et pouvoir éventuellement le faire de manière dynamique, tenant compte des ressources réellement disponibles à un instant *t*. Pour cela, la description d'une ressource doit permettre la mise en correspondance entre les services sollicités (et donc à exécuter) et les ressources disponibles, dans toute leur variété.

Afin de permettre ce placement, deux informations nous semblent primordiales : les *capacités* Ca_i d'une ressource et ses *contraintes* Co_i. Une ressource a des capacités d'exécution, en rapport aussi bien aux fonctionnalités qu'elle est capable d'exécuter, qu'à l'environnement d'exécution qu'elle offre. Prenons, par exemple, une machine virtuelle, elle offre un certain environnement pour l'exécution de services, lequel peut être caractérisé par un contexte d'exécution : nombre de cœurs, type de processeur, mémoire disponible, capacité de stockage, interfaces réseaux, etc. ; alors qu'un collaborateur peut être vu aussi à travers les fonctionnalités qu'il est autorisé à réaliser, en plus d'avoir un contexte qui lui est associé (*e.g.* sa localisation, les rôles qu'il peut jouer dans l'organisation, son niveau d'expertise...). L'équation (3) résume ainsi les capacités d'une ressource. Celles-ci sont vues à travers l'ensemble de fonctionnalités $\mathcal{F}$ dont une ressource est reconnue comme étant capable de réaliser, et à travers le contexte d'exécution $\mathcal{CXE}$ qu'elle offre. Il convient de signaler qu'en absence de fonctionnalités (c.a.d. un ensemble $\mathcal{F}$ vide), on considère qu'il n'y a aucune limitation connue ou souhaitée sur les fonctionnalités qu'une ressource est capable d'exécuter.

$$Ca_i = < \mathcal{F}, \mathcal{CXE} > \tag{3}$$

Les contraintes se présentent comme des exigences non-fonctionnelles, de nature notamment organisationnelle, propres à une ressource. Par exemple, un certain dispositif (*e.g.* un Raspberry Pi) peut avoir son usage limité à un certain département (pour des raisons de sécurité). En d'autres termes, l'usage de cette ressource impose au service souhaitant l'utiliser d'appartenir à ce département précis. Il s'agit donc d'une contrainte que la ressource impose aux services souhaitant l'utiliser. A cet ensemble de contraintes Co_i s'ajoute un ensemble de propriétés $\mathcal{P}r$, décrivant les propriétés d'une ressource. Ces propriétés peuvent être vues comme de métadonnées complétant la description d'une ressource sur différents aspects, tel que le positionnement de la ressource dans l'organisation (par exemple, l'unité d'appartenance de la ressource, son année d'acquisition/entrée dans le système, etc.). En outre, une ressource possède, tout comme les services, un contexte $\mathcal{CX}$ dans lequel elle se trouve, permettant ainsi de décrire une ressource r_i, à un instant *t*, comme indique l'équitation (4) :

$$r_i^t = < Ca_i, Co_i, \mathcal{P}r, \mathcal{CX} > \tag{4}$$

Afin de permettre une réelle réflexion sur l'usage des ressources, voir leur gestion de manière automatisée, il est également nécessaire de revoir la description des services pour qu'une mise en correspondance entre les services à exécuter et les ressources disponibles soit possible. Dans la description d'un service représentée dans l'équation (1), on dispose d'un contexte requis par le service par rapport à son client. Il s'agit d'indiquer une situation favorable (côté client) pour la réussite du service. Cependant aucun élément dans cette définition permet d'indiquer les possibles contraintes d'un service côté serveur. L'hypothèse sous-jacente est que la ressource devant exécuter le service existait déjà et lui était réservée au préalable. C'est cette hypothèse qui est remise en cause avec l'évolution des ressources dans les SIP. Il devient ainsi nécessaire de revoir cette description d'un service en lui ajoutant un ensemble de contraintes $\mathcal{CoR}$ imposées par ce service aux ressources pouvant assurer son exécution.

Puis, les ressources pouvant elles-mêmes avoir des contraintes, comme l'indique l'équation (4), il serait également opportun de permettre une description plus complète de ces services. Pour cela, un ensemble de propriétés $\mathcal{Pr}$ est ajoutée à la description d'un service. L'objectif de cet ensemble est de permettre l'indication de propriétés autres que les fonctionnalités et l'intention dans la description d'un service. Comme les propriétés décrivant les ressources, celles-ci peuvent être vues comme des métadonnées décrivant les services sous plusieurs aspects, dont les aspects organisationnelles (*i.e.* positionnement du service dans l'organisation). Ces informations peuvent ainsi être utilisées pour limiter le placement de ces services sur certaines ressources à travers les contraintes associées à celles-ci.

On arrive ainsi à trois ensembles qui vont définir la notion d'espace de services, illustrées à travers la Figure 1 : un ensemble $\mathcal{R}^t$ de ressources disponibles à un instant t, représenté dans l'équation (5) ; un ensemble $\mathcal{A}^t$ d'entités actives (*i.e.* les services) proposés par le système à un instant t, visible dans l'équation (6) ; et un ensemble $\mathcal{P}^t$ d'entités passives (*i.e.* les capteurs), symbolisées par l'équation (7), représentant les capteurs permettant l'observation de l'environnement. Ces trois ensembles forment l'espace de services, représenté dans l'équation (8), lequel est un espace composé d'entités pouvant être soit des entités actives, représentant les services offerts par le système ; soit des entités passives, représentant les capteurs utilisés pour observer le contexte ; soit des ressources utilisées pour exécuter les services dans un contexte donné. Il est important d'observer que chaque entité, soit elle active (sv_i), passive (cp_i) ou une ressource (r_i^t), représente un rôle joué par un élément présent, à un instant t, dans le système. Par exemple, dans la Figure 1, la ressource r_3, au centre de l'image, peut représenter un RaspberryPI exécutant un service sv_3 et permettant l'observation de l'humidité de l'air (cp_6). Du même, la ressource r_2 permet d'observer plusieurs informations sur l'utilisateur (GPS, préférences), étant ainsi perçue non seulement non seulement comme une ressource, mais aussi comme une entité passive. Un même élément peut ainsi être perçu à travers les différents rôles qu'il joue dans le système, offrant une abstraction totale sur la vraie nature de l'élément. Nous avons de ce fait un partage de responsabilités entre les rôles, et donc, les entités : les ressources permettent l'exécution des entités actives (services), les entités passives (capteurs) sont là pour observer ces ressources et les entités actives.

$$\mathcal{R}^t = \{ r_i^t \}, \text{ où } r_i^t = < Ca_i , Co_i , C\mathcal{X} > \quad (5)$$

$$\mathcal{A}^t = \{ sv_i \}, \text{ où } sv_i = < I , F, C\mathcal{X}\mathcal{R}, Co\mathcal{R}, Pr, C\mathcal{X} > \quad (6)$$

$$\mathcal{P}^t = \{ cp_i \}, \text{ où } = \{ O_{cpi} , C\mathcal{X} \} \quad (7)$$

$$\xi^t = \{ e_i \mid e_i \in \mathcal{A}^t \vee e_i \in \mathcal{P}^t \vee e_i \in \mathcal{R}^t \} \quad (8)$$

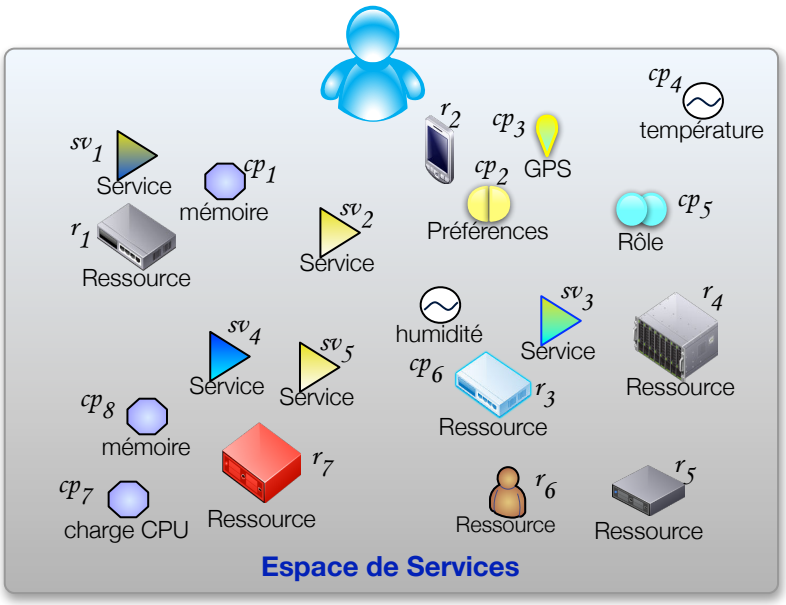


Figure 1. Représentation d'un espace de services avec ses composants.

On observe à travers les équations (5) à (8) que, contrairement à la proposition originelle (Kirsch Pinheiro *et al*., 2013), toutes ces définitions prennent désormais en considération l'aspect temporel : il s'agit de la vision d'un ensemble d'entités à un instant *t*. A travers cette vision temporelle, c'est la dynamicité de l'espace de service qui est mise en avant, avec des entités qui peuvent intégrer cet espace et d'autres qui peuvent disparaître. A travers cet aspect, c'est aussi la notion de disponibilité qui apparaît. On parle de ressources disponibles, de services disponibles et surtout, on admet dès le départ que ces ensembles vont évoluer au fil du temps. Cette idée d'évolution est une condition *sine quoi non* pour la prise en compte des ressources, puisque c'est le dynamisme de ces ressources qui motive en grand partie qu'elles soient désormais considérées dans la conceptualisation des SIP.

L'espace de services ainsi revu représente un cadre conceptuel permettant la conceptualisation d'un SIP à travers ces trois éléments fondateurs : les services, les ressources et le contexte. Ce cadre permet d'abstraire la complexité et l'hétérogénéité de l'environnement, et notamment des ressources qui l'intègrent. Celles-ci ont ainsi été intégrées à ce cadre, permettant désormais une analyse de ces éléments de manière transparente (c.a.d. indépendante des technologies impliquées).

4. Illustration

Afin d'illustrer l'intérêt de nous propos, nous allons considérer le cas d'un aéroport en guise d'exemple. Les organisations devant gérer ce genre d'infrastructure de transport sont souvent dotées d'un SI plutôt complexe dans lequel multiples technologies, partenaire et contraintes cohabitent. Ces SI sont amenés à proposer différents services à leurs consommateurs, mais aussi à leurs propres employés et à leurs partenaires (compagnies aériennes, prestataires, fournisseurs, etc.). Ces derniers peuvent proposer à leur tour des services qui peuvent (ou doivent) être intégrés au système. A ces services de nature IT, déjà très hétérogènes, s'ajoute une forte intégration à l'environnement physique, avec la gestion de halls et hangars, tapis, systèmes et portiques de sécurité, etc., qui rajoutent d'autres services mais également des multiples capteurs. En effet, l'observation de l'environnement physique faut de plus en plus partie du panorama de ces infrastructures : par exemple, l'observation de la température dans les halls et salles d'embarquement, du trafic passager dans les zones de contrôle, du flux bagage dans les tapis, des taux d'occupation des parkings, le volume passager dans les zones d'attente, etc. Toutes ces informations doivent être remontées grâce à l'intégration de capteurs divers et variés (capteurs de température, de mouvement, de passage, mais aussi cameras, dispositifs réseaux…).

A ces éléments s'ajoutent des ressources pour l'exécution des services tout aussi variées. On peut compter sur l'existence de serveur ou *data centers* propres à l'organisation, mais également sur la mise à disposition de ressources par des partenaires (notamment pour l'exécution de service proposés par ceux-ci), et sur des ressources qu'on peut louer sur des infrastructures cloud. A celles-ci, plus traditionnelles, s'ajoutent les ressources intégrées directement sur l'environnement qui peuvent aussi contribuer à l'exécution de certains services (*e.g.* ressources réseaux, IoT...). La prise en compte des différentes ressources pour l'exécution de services va également entrainer l'observation d'autres informations liées au contexte d'exécution de ces ressources (consommation électrique, indicateurs de performance, bande passante, etc.), notamment à travers l'usage d'outils de monitoring.

Une prise en main globale d'une telle variété d'éléments s'avère une tâche plutôt complexe. Des multiples décisions doivent être prises aussi bien d'un point de vu business que technique : quels services proposer ? dans quelles conditions ? où exécuter ces services ? quelles informations observer pour pouvoir les garantir ? Par ailleurs, certaines décisions peuvent être planifiées et programmées à l'avance, alors que d'autres viennent en réaction à l'état courant du système, voir à l'influence des facteurs extérieurs à celui-ci (grèves, tempêtes, retards, etc.). Par exemple, le placement de certains services (peut-être sous la forme de microservices) sur le cloud

peut relever aussi bien d'une décision purement business (*e.g.* réduire les coûts d'exploitation), mais aussi technique (*e.g.* éviter la surcharge ou contourner la panne d'un serveur). Pareillement, un afflux trop important de passagers peut rapidement surcharger l'infrastructure réseau et ainsi impacter le transfert des informations issues des capteurs ou l'usage des ressources réseau pour le prétraitement de certaines informations (scénario typiquement invoqué par le *fog computing*).

L'usage de la notion d'espace de service permettrait ici de simplifier la prise en main de ces systèmes par la conceptualisation de multiples espaces où différents services, ressources et éléments observables coexistent. On peut donc imaginer chaque hall, salle d'embarquement ou zone de l'aéroport comme un espace de services, et ainsi conceptualiser, en fonction de l'usage attendue, les services qui y seront proposés (par leurs fonctionnalités et leurs intentions) et dans quelles conditions ils le seront (le contexte requis). On peut également considérer les ressources qui pourront être utilisés pour l'exécution de ces services (*e.g.* machines virtuelles sur une infrastructure cloud ou sur un *data center*, un nanordinateur de type RaspberryPi intégré à l'environnement, ou même des terminaux portables, notamment dans le cas de services proposés sous la forme d'application mobiles). On va également pouvoir considérer les informations qui pourront être observées à partir de l'environnement et des ressources (*e.g.* température, humidité, nombre de passagers, charge CPU, bande passante, mémoire disponible, etc.). Une telle conceptualisation permettrait une meilleure gestion de ces espaces d'abord en amont, d'un point de vue business, mais également en aval, pour l'adaptation du système pendant son exécution. A travers notamment les ressources, on observe que la notion d'espace de service ne se limite pas à des éléments disposés dans une zone géographique précise, mais à une véritable conceptualisation au service du SI. Par exemple, on pourrait également conceptualiser des espaces des secteurs en fonction des secteurs d'activités de l'aéroport (accueil passager, traitement bagages, maintenance, sécurité…) ou encore du public visé (employé, consommateurs, prestataires…).

Il s'agit avant tout d'un cadre de réflexion permettant une meilleure prise en main des nouveaux SI de manière transparente, sans tenir particulièrement compte des différences technologiques entre les éléments considérés. Un décideur peut ainsi conceptualiser plus l'aéroport et les services proposés par son SI, réfléchir aux ressources qui seront mis à disposition de ce système et les informations qu'on pourra observer à partir de celui-ci, sans pour autant maîtriser tous les aspects techniques impliquant tous ces éléments.

5. Conclusions

Le choix de placement d'un service sur une certaine ressource dans les SI devient stratégique au fil et à mesure que le nombre et la nature des ressources pouvant être utilisées augmente et se diversifie. Il devient nécessaire de réfléchir au rôle de ces ressources dans les futurs SIP, lesquels se caractérisent justement par l'hétérogénéité et la dynamicité de leur environnement et de leurs ressources.

Dans cet article nous abordons cette question en proposant d'intégrer la notion de ressource dans la conceptualisation des SIP. Nous avons ainsi étendu la notion d'espace de services afin d'y représenter de manière explicite les ressources disponibles dans un SIP. Ce cadre conceptuel permet la conceptualisation de ces systèmes tout en faisant abstraction des technologies qui les caractérisent. Il s'agit avant tout d'un outil de conception offrant un langage unifié, aussi bien au niveau métier qu'au niveau technique. En effet, aussi bien au niveau technique, qu'au niveau métier, il est possible d'exprimer les services offerts par le système, les ressources disponibles pour l'exécution de ces services et de capteurs envisageables pour l'observation de l'environnement.

Ce cadre propose ainsi un langage commun entre ces niveaux, qui peuvent ainsi échanger autour de ces concepts. La représentation d'un espace de service permet au niveau métier de réfléchir aux services qui seront exécutés au niveau technique, et à ce dernier de remonter de l'information sur l'état du système au premier, lequel peut donc réagir et réfléchir à des possibles adaptations au niveau technique (voir Figure 2). L'espace de services se présente ainsi comme un outil d'aide à la décision pour le niveau métier, et un outil pour l'adaptation pour le niveau technique.

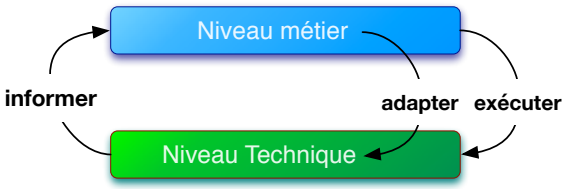


Figure 2. Interaction entre les niveaux métiers et technique.

Cependant, pour que l'application de ce cadre devienne plus aisée, et qu'ainsi son rôle de langage commun puisse être pleinement atteint, un important travail de méthodologie reste à faire. En effet, la définition d'espace de services originelle était accompagnée d'une méthodologie d'application (Najar et al., 2014), qui doit être révisée et complétée pour la prise en compte des ressources, en plus des services, à ces deux niveaux (métier, technique). Celle-ci devra tenir compte particulièrement de la dynamique et de la cohérence entre les niveaux, ce qui représente, à nous yeux, un challenge majeur pour la gestion des futurs SIP.

Bibliographie

Alhamazani, K., Ranjan, R., Mitra, K., Rabhi, F., Jayaraman, P. P., Khan, S. U., Guabtni, A., Bhatnagar, V. (2015). An overview of the commercial cloud monitoring tools: research dimensions, design issues, and state-of-the-art. *Computing*, vol. 97, n° 4, p. 357-377.

Bessai K. (2014). *Gestion optimale de l'allocation des ressources pour l'exécution des processus dans le cadre du Cloud*. Université Paris1 Panthéon-Sorbonne, 2014.

Da Cunha Rodrigues, G., Calheiros, R. N., Guimaraes, V. T., Santos, G. L. d., de Carvalho, M. B., Granville, L. Z., Tarouco, L. M. R., Buyya, R. (2016). Monitoring of Cloud Computing Environments: Concepts, Solutions, Trends, and Future Directions. *Proceedings of the 31st Annual ACM Symposium on Applied Computing (SAC 2016)*, ACM, p. 378-383.

Chen S., Zhang T., Shi W. (2017). Fog Computing. *IEEE Internet Computing*, vol. 21, n° 2, p. 4-6.

Gubbi J., Buyya R., Marusic S., Palaniswami M. (2013). Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, vol. 29 n° 7, p. 1645-1660.

Estublier J., Vega G., Damou E. (2012). Resource Management for Pervasive Systems. Ghose A., Zhu H., Yu Q., Delis A., Sheng Q. Z., Perrin O., Wang J., Wang Y. (Eds.), *Service-Oriented Computing - ICSOC 2012 Workshops*, Lecture Notes in Computer Science, vol 7759, Springer, p. 368-379.

Kirsch Pinheiro M., Le Grand B., Souveyet C., Najar S. (2013) Espace de Services : Vers une formalisation des Systèmes d'Information Pervasifs, *XXXIème Congrès INFORSID 2013 : Informatique des Organisations et Systèmes d'Information et de Décision*, 29-31 Mai 2013, Paris, France, p. 215-223.

Kourouthanassis P. E., Giaglis G. M. (2006). A Design Theory for Pervasive Information Systems. *3rd Int. Workshop on Ubiquitous Computing 2006*, Paphos, Cyprus, p. 62-70.

Kourouthanassis P. E., Giaglis G. M., Karaiskos, D. C. (2010). Delineating `pervasiveness' in pervasive information systems: a taxonomical framework and design implications. *Journal of Information Technology*, vol. 25, n° 3, p. 273-287.

Maurer M., Brandic I., Sakellariou R. (2012). Self-Adaptive and Resource-Efficient SLA Enactment for Cloud Computing Infrastructures. *IEEE Fifth International Conference on Cloud Computing*, p. 368-375.

Mell P., Grance T. (2011), *The NIST Definition of Cloud Computing*. Recommendations of the National Institute of Standards and Technology. NIST Special Publication 800-145. https://doi.org/10.6028/NIST.SP.800-145.

Mulfari D., Fazio M., Celesti A., Villari M., Puliafito A. (2015). Design of an iot cloud system for container virtualization on smart objects. *3rd Workshop on CLoud for IoT (CLIoT 2015)*, in conjunction with the *European Conference on Service-Oriented and Cloud Computing (ESOCC 2015)*, *Communications in Computer and Information Science* (CCIS), vol. 567, Springer, p.33-47.

Najar S., Kirsch Pinheiro M., Le Grand B., Souveyet C. (2014), A user-centric vision of service-oriented Pervasive Information Systems. *8th International Conference on Research Challenges in Information Science*, p. 359-370.

Parashar M., Pierson J.-M. (2010). Pervasive Grids: Challenges and Opportunities. In: Li K.-C., Hsu C.-H., Yang L.T., Dongarra J., Zima H. (Eds.), *Handbook of Research on Scalable Computing Technologies*, IGI Global, p. 14-30.

Rao B., Angelov B. (2009). Pervasive Information Systems Value Chain: A Services Perspective. *International Journal of Innovation and Technology Management*, vol. 06, n° 01, p. 17-40.

Singh S., Chana I. (2016). A Survey on Resource Scheduling in Cloud Computing: Issues and Challenges Journal of Grid Computing, vol. 14, n° 2, p. 217-264.

Steffenel L., Kirsch-Pinheiro M. (2015). When the Cloud goes Pervasive: approaches for IoT PaaS on a mobiquitous world. *EAI International Conference on Cloud, Networking for IoT systems (CN4IoT 2015)*, LNICST, vol. 169, Springer, p. 347-356.

Tyagiand R., Gupta, S.K. (2018). A survey on scheduling algorithms for parallel and distributed systems. Mishra A., Basu A., Tyagi V. (eds), *Silicon Photonics & High Performance Computing. Advances in Intelligent Systems and Computing*, vol. 718, p. 51–64.

Villari M., Fazio M., Dustdar S., Rana O., Ranjan R. (2016). Osmotic Computing: A New Paradigm for Edge/Cloud Integration. *IEEE Cloud Computing*, vol. 3, n° 6, p. 76-83.

Wang T, Wei X, Liang T, Fan J (2018). Dynamic tasks scheduling based on weighted bi-graph in mobile cloud computing. *Sustainable Computing: Informatics and Systems*, vol. 19, p. 214–222.

DEMOS : une méthode de conception participative pour un empowerment démocratique des utilisateurs de SI

Raphaëlle Bour[1], Nathalie Vallès-Parlangeau[1], Chantal Soule-Dupuy[1]

1. IRIT, Université Toulouse 1 Capitole
21 Allées de Brienne, 31015 Toulouse, France
prenom.nom@ut-capitole.fr

RESUME. La question de la démocratie est au cœur de nos préoccupations sociétales actuelles. Les organisations, dans toutes leurs formes, ne peuvent s'affranchir de cette problématique qui représente un véritable enjeu d'empowerment pour leurs salariés. Le système d'information (SI) des organisations joue un rôle central sur ce point. En effet, si l'on s'accorde à considérer qu'il est le langage de l'organisation, alors il est celui qui encapsule les valeurs, les normes : autant d'éléments qui doivent être débattus avec les utilisateurs. Les démarches de conception participative proposent d'instaurer ce débat, leur conférant ainsi un caractère plus démocratique que d'autres méthodes de conception traditionnelles. Nous proposons d'aller plus loin en présentant une démarche de conception certes démocratique, mais qui permet également l'intégration de la démocratie au sein du SI lui-même. Cette méthode nommée DEMOS est l'objet de ce papier. Elle sera décrite et son déroulement sera illustré par des éléments concrets provenant d'une expérimentation menée avec le service de formation continue de l'Université.

ABSTRACT. The issue of democracy in society is at the heart of our current concerns. Organizations are also concerned by this issue. It is a real challenge of empowerment for employees. The Information System (IS) is the language of the organization, it requires a debate about norms and values to be considered as democratic. Participatory design proposes to debate with users, it is a kind of democratic approach. We propose to go further by integrating democracy into IS. We present in this article a method called DEMOS. We describe it with an instantiation by a real experiment provided by a "lifelong training" service at the University. All aspects of the method are addressed: from requirement engineering phase to implementation..

Mots-clés : Démocratie, Ingénierie des méthodes, Conception de système d'information, Recueil des exigences, Point de vue, Utilisateurs finaux, Conception participative, Agilité.

KEYWORDS: Democracy, Method engineering, Information system design, Requirement engineering, Viewpoint, End-users, Participatory design, Agility

1. Introduction

Le Système d'Information ne reflète pas la réalité, il la construit. Les valeurs d'une organisation, les normes qu'elle met en place sont le fruit de choix que le système d'information implémente au travers des nomenclatures, des modes de comptage, des indicateurs, et bien sûr du vocabulaire utilisé. A ce sujet, Brey parle de « valeurs embarquées » (Brey, 2010) auxquelles Mingers attribue un « impact moral » (Mingers and Walsham, 2010). Les concepteurs et constructeurs de systèmes d'information implémentent parfois des standards sans même réaliser qu'ils répondent à des choix stratégiques en termes de normes et de relations de pouvoir dans l'organisation (Walsham, 2012). Depuis quelques années, des auteurs s'intéressent à cette question et cherchent des moyens d'éliciter ces valeurs embarquées (Floridi, 2010) et de les prendre en compte lors de la construction de systèmes d'information. Salles et Colletis ont proposé en 2008 une grille de lecture à trois niveaux permettant de faire le lien entre représentations, modèles et normes (Salles and Colletis, 2008), c'est-à-dire entre le niveau le plus macroscopique d'une organisation, fait de sa vision du monde, et son incarnation dans le système d'information au travers du vocabulaire et des codifications qui y sont implémentées (Salles, 2015a).

La question de la démocratie, vaste sujet de l'éthique informatique, est un prolongement de ces premières réflexions au sujet des valeurs. On peut considérer dans un premier temps que la démocratie se traduit par le fait d'instaurer des débats, des discussions au sujet de ces valeurs et de ces normes. Pour être démocratique, ce débat se doit d'impliquer tous ceux qui sont affectés par la mise en place du système d'information, utilisateurs finaux en tête. En effet, ceux-ci sont les premiers impactés au quotidien par les choix de normalisation visibles au travers du système d'information : vocabulaire utilisé, procédures mises en place, etc. Ils ont à ce titre un droit de participation à la conception de « leur » système d'information, pour que ce processus se passe de manière démocratique. Simonsen va même jusqu'à qualifier de « droit humain de base » le fait, pour un utilisateur, d'avoir l'opportunité d'influencer le processus de conception et d'implémentation s'il est affecté par les changement que cela induit (Simonsen, 2013). Mais la question de la démocratie ne se limite pas à cela. En effet, un processus démocratique de conception de système d'information ne saurait garantir un système d'information lui-même démocratique. Pour cela, le système d'information doit respecter une autre facette de la définition de la démocratie : « la garantie de l'accès à une pluralité de points de vue » (Salles, 2015b). En d'autres termes, un système d'information démocratique est un système qui respecte et prend en compte les différents points de vue, ne se conformant pas alors à un point de vue dominant. Dans le cadre de notre étude à l'Université par exemple, un système d'information démocratique serait celui qui prend en compte le point de vue des gestionnaires, de l'administration, mais aussi celui des équipes pédagogiques.

Pour reprendre les termes de Van den Hoven, nous proposons une « intégration proactive » (Van den Hoven, 2008) de démocratie avec notre méthode de conception de système d'information DEMOS (pour DEsign Method for demOcratic

information System). Répondant aux définitions du paragraphe précédent, cette méthode intègre la démocratie de deux façons :

- un processus de conception démocratique, qui permet aux utilisateurs de débattre au sujet des normes, des valeurs et des points de vue qui façonnent le SI,

- un système d'information démocratique, qui respecte les points de vue exprimés et les implémente.

Dans un premier temps nous présentons un état de l'art portant sur l'implication des utilisateurs dans les démarches de conception. Nous introduisons dans un second temps la méthode DEMOS : quelles sont les problématiques qu'elle adresse ? Quelles sont les intentions de la méthode ? Quelles stratégies pour y répondre ? Puis nous nous attardons sur la notion de point de vue au sein de la méthode, et sur la façon dont ce concept clé est pris en compte dans le métamodèle. Nous déroulons ensuite le processus dans son ensemble, en l'illustrant avec une expérimentation menée au sein du service de formation continue de l'université. Enfin, nous proposons les premiers résultats d'évaluation de la méthode.

2. Etat de l'art : l'implication des utilisateurs dans les démarches de conception

L'implication grandissante des utilisateurs dans le processus de conception répond à un problème majeur des systèmes d'information : leur difficulté d'adoption par ces mêmes utilisateurs (McConnell, 2006). Le fait d'impliquer les utilisateurs vient accroître les chances d'adéquation du système aux besoins, et améliorer ainsi ses qualités fonctionnelles. Mais depuis quelques années, et notamment avec l'émergence des méthodes agiles, cette intégration des utilisateurs au processus de conception a aussi pour but de garantir un « empowerment démocratique » de ces derniers via une participation directe à la prise de décision (Kautz, 2011). Plusieurs niveaux d'implication des utilisateurs sont évoqués dans la littérature : de la conception centrée utilisateurs à la conception participative en passant par la conception coopérative. L'utilisateur passe tantôt du statut d'objet d'étude à celui de leader du processus (Andre and Christian, 2016). Les méthodes agiles proposent elles aussi d'intégrer l'utilisateur au processus de développement et/ou de conception. Cet état de l'art se propose d'établir un panorama de ces différentes méthodes.

La conception centrée utilisateurs intègre les utilisateurs finaux dès le début du processus. Le challenge consiste ici à comprendre au mieux leurs besoins, afin de concevoir un système qui y réponde. Les équipes de conception observent les utilisateurs, leurs interactions, et tirent parti des résultats de leurs analyses en améliorant l'utilisabilité du système (Schwartz et al., 2009). Mais si les utilisateurs sont le point de focus des concepteurs, comme le dit Ferrario : les développeurs continuent d'être les meneurs du projet (Ferrario et al., 2014). Les utilisateurs sont simplement consultés mais ne prennent pas part aux décisions, ce qui ne garantit donc en rien un apport de démocratie dans le processus (Darses, 2004).

La conception coopérative se distingue de la conception centrée utilisateurs au sens où elle cherche à faire véritablement coopérer les utilisateurs et les concepteurs

(Greenbaum, 1991). Les utilisateurs interagissent avec un prototype du système en cours de développement et peuvent induire des modifications dans le développement dudit système via les remarques et appréciations qu'ils expriment. Ici encore, bien que l'utilisateur coopère d'avantage, il n'est pas impliqué dans les choix de conception en amont (discussions autour des normes, des principes du système) et son rôle reste consultatif (Darses et al., 2001).

Les méthodes de conception participative sont sans doute celles où l'implication des utilisateurs finaux est la plus grande. Cette participation des futurs utilisateurs est d'ailleurs considérée comme le principe central de ces méthodes selon Kensing et Bloomberg (Kensing and Blomberg, 1998). L'utilisateur n'est plus un sujet d'étude, mais un partenaire à part entière (Dell'Era and Landoni, 2014), qui coconçoit son système (Sanders and Stappers, 2008) en participant à un processus créatif (Mahaux et al., 2013). Cette fois-ci, le but est non seulement d'améliorer la qualité du système (Damodaran, 1996), mais également de garantir aux utilisateurs un empowerment démocratique grâce au haut niveau de participation à la prise de décision qui leur est proposé (Kujala, 2003). L'utilisateur est la source de connaissance du concepteur (Simonsen, 2013) et évalue le système jusqu'à son développement. Pour ces raisons, certains auteurs comme Sanders qualifient ces méthodes de démocratiques (Sanders and Stappers, 2008).

Les méthodes agiles sont elles aussi considérées comme des approches participatives. Dans les faits, l'agilité et la conception participative partagent les mêmes valeurs, comme le décrit le manifeste Agile : privilégier les individus et leurs interactions, favoriser la collaboration, permettre l'adaptation au changement… Cela étant, les méthodes agiles sont avant tout des méthodes de développement, et peu d'entre elles abordent les champs amont de la conception : recueil des exigences, analyse des besoins. Notons comme exception la méthode RAD, qui au travers de sa phase de cadrage invite les utilisateurs à spécifier leurs exigences. Cette méthode est sans doute la plus « participative » de toutes les méthodes agiles (Millington and Stapleton, 1995). Pour SCRUM et XP, les deux autres méthodes agiles les plus courantes, l'implication des utilisateurs n'est pas si évidente. En effet, si XP évoque le fait que les utilisateurs participent à l'expression des besoins, cette méthode s'attache principalement à décrire les bonnes pratiques de développement et l'implication d'un client au processus (Chamberlain et al., 2006; Sharp and Robinson). Ce terme de client se distingue bien entendu de celui d'utilisateur. SCRUM propose elle d'intégrer au processus un Product Owner, qui peut être considéré comme un représentant des clients et des utilisateurs, garant du retour sur investissement. On voit au travers de ces deux méthodes que c'est avant tout le client qui est impliqué dans les méthodes agiles (Highsmith and Cockburn, 2001) et que cela ne permet donc en rien de garantir un empowerment démocratique des utilisateurs finaux.

3. La méthode DEMOS

3.1. Présentation générale de la méthode

DEMOS est une méthode de conception de système d'information démocratique. Nous avons identifié quatre principes que cette méthode se doit d'appliquer afin de garantir le respect des objectifs fixés : un processus démocratique de conception, ainsi que la production d'un système d'information démocratique. Ces quatre principes sont les suivants :

- **impliquer** les utilisateurs finaux dans un processus participatif et démocratique,

- favoriser un **débat** démocratique permettant l'émergence de points de vue,

- concevoir un système d'information démocratique qui prend en compte ces **points de vue,**

- assurer une traçabilité des points de vue dans **l'évolution** du système durant son cycle de vie.

DEMOS est représentée sous la forme d'une MAP (figure 1) : une « structure navigationnelle » développée par Rolland (Rolland et al., 1999). Ce formalisme permet de décrire la méthode comme un ensemble d'intentions (ovales) et de stratégies (flèches) pour les atteindre. DEMOS présente 5 intentions et 9 stratégies associées. Des ateliers successifs rythment le processus itératif : ainsi plusieurs stratégies rétroactives permettent de venir enrichir une intention précédente (exemple de la stratégie n°8 sur la figure 1).

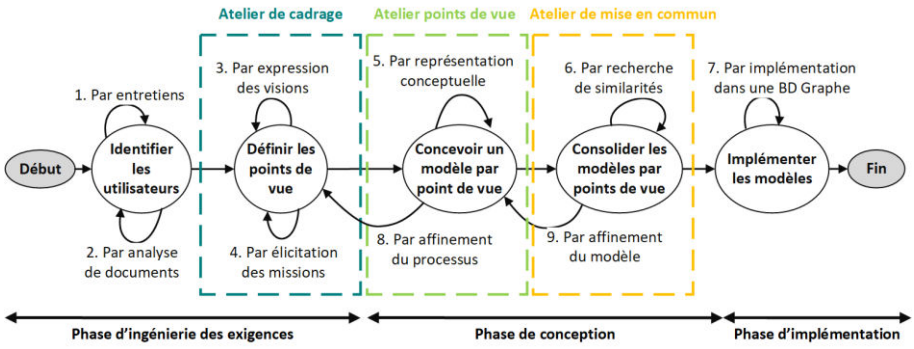


Figure 1. MAP générale de la méthode DEMOS

Les intentions et leurs stratégies présentées dans la MAP permettent de garantir les quatre principes de la méthode. Nous présentons dans le tableau 1 la correspondance entre les intentions et chacun de ces principes.

Principes de la méthode	Intention de la méthode
Impliquer les utilisateurs finaux dans un processus participatif et démocratique	1ère intention : **Identifier les utilisateurs finaux**
Favoriser un débat démocratique permettant l'émergence de points de vue	2ème intention : **Définir les points de vue**
Concevoir un système d'information démocratique qui prend en compte ces points de vue	3ème intention : **Concevoir un modèle par point de vue** 4ème intention : **Consolider les modèles par point de vue**
Assurer une traçabilité des points de vue dans l'évolution du système durant son cycle de vie	5ème intention : **Implémenter les modèles**

Tableau 1. Correspondance entre les intentions et les principes de DEMOS

3.2. Le concept clé de point de vue au cœur du métamodèle

La présentation générale de la méthode DEMOS met en lumière un concept central, au cœur des trois principales intentions de la méthode : le point de vue. Comme nous l'avons expliqué, l'émergence, le respect et la traçabilité des points de vue sont une garantie du respect des objectifs de démocratie qui sont poursuivis. Cette notion peut être explicitée en suivant deux questionnements distincts : au sein du groupe des utilisateurs, qui sont ceux qui partagent un point de vue commun, et donc à contrario qui sont ceux qui ont des points de vue distincts ? Qu'est-ce qui constitue le point de vue ?

Pour répondre à la première question, établissons d'ores et déjà une première distinction entre les notions de point de vue, d'utilisateur, de rôle ou de métier. Le système d'information est conçu et construit pour répondre aux besoins de plusieurs ***utilisateurs***, i.e. les futurs usagers du système d'information, directs ou indirects. Ces utilisateurs sont regroupés sous une certaine nomenclature : celle des métiers. Ces ***métiers*** désignent les professions de chacun des utilisateurs au sein de l'organisation, et le niveau hiérarchique auquel cela renvoie. A chaque métier, des missions sont assignées. On entend par ***missions*** les différentes tâches regroupées dans la fiche de poste de l'utilisateur, ainsi que les tâches qu'il effectue plus « officieusement ». Ces missions sont assurées via la participation à des ***processus*** métier, décomposés en ***activités***. Les utilisateurs du système d'information endossent des ***rôles*** particuliers dans chacun de ces processus métier. Mais certains utilisateurs, bien que ne partageant pas les mêmes métiers, peuvent répondre à un même cadre normatif. On entend par cadre normatif un ensemble composé des normes et d'une vision du domaine métier correspondant à une stratégie commune, à des objectifs partagés. C'est ce cadre normatif qui constitue la notion de ***point de vue***, sur laquelle repose le métamodèle de DEMOS présenté en figure 2. La notion de point de vue émerge tantôt de façon évidente, tantôt d'une manière plus subtile. C'est la raison pour laquelle différentes stratégies croisées sont nécessaires pour définir les points de vue d'un domaine métier : expression de visions du ***domaine métier*** par les utilisateurs, élicitation des missions de chacun. C'est l'objet de la deuxième intention de la méthode DEMOS, qui sera décrite en détail dans la partie suivante.

Qu'est-ce qui constitue le point de vue ? Si l'on s'accorde maintenant sur le fait que les points de vue sont en lien direct avec les cadres normatifs, i.e. les visions du domaine métier, alors on peut affirmer qu'ils induisent l'utilisation d'un vocabulaire différent. En effet, et comme nous l'avons évoqué en introduction, les termes employés par les utilisateurs, leurs éléments de langage sont très étroitement liés à la vision qu'ils portent sur le domaine métier, et aux normes et valeurs qui l'accompagnent. Le vocabulaire est composé de plusieurs éléments : les ***concepts***, i.e. les représentations d'entités du monde réel, les ***descripteurs*** qui permettent de décrire les concepts ainsi que les ***liens*** entre concepts. Bien que chaque point de vue ait son propre vocabulaire, les utilisateurs d'un même système d'information ont en commun des processus, et il est donc nécessaire de dresser des ponts entre les concepts de chacun des points de vue, ceci afin d'identifier les concepts et descripteurs qui sont similaires. Ce sont les ***liens de similarité*** entre descripteurs qui assurent cette mise en cohérence. La figure 2 présente le métamodèle de DEMOS explicitant cette notion de vocabulaire par point de vue. Le vocabulaire associé à chacun des points de vue nécessite d'être représenté sous la forme d'un modèle conceptuel par les utilisateurs. C'est l'objet des troisièmes et quatrième intentions de la méthode, présentées dans la partie suivante de l'article

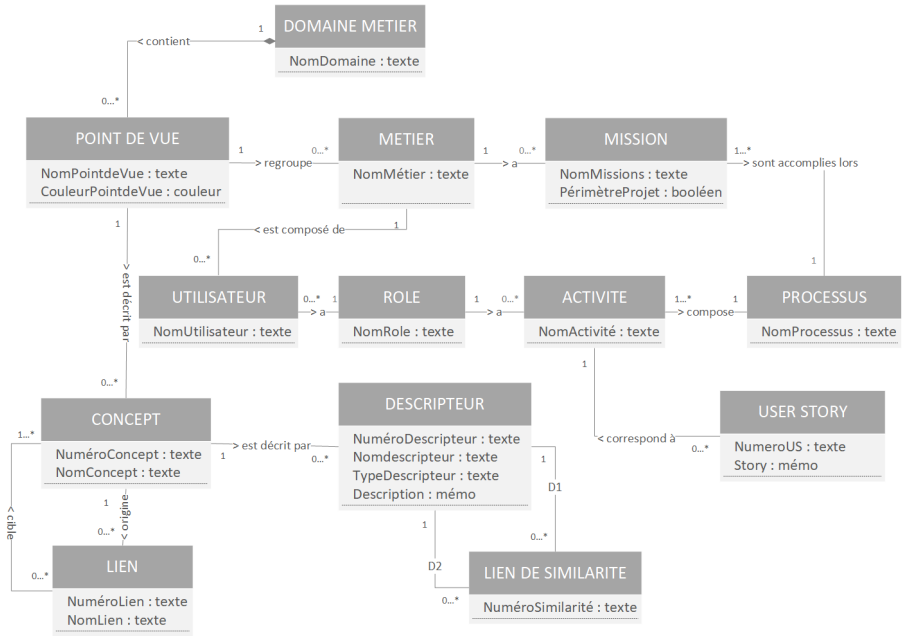


Figure 2. Métamodèle DEMOS : les points de vue dans le domaine métier

3.3. Le processus DEMOS détaillé et instancié

Cette troisième partie nous permet de décrire plus en détail les différentes intentions et stratégies de DEMOS. Nous nous concentrons sur les trois intentions

centrales de la méthode (cf. figure 1) qui concernent la définition, la modélisation et la consolidation des points de vue. Notre description prend la forme de sous-MAP décrivant les sections de la méthode. Une section s'écrit de la façon suivante : <*Intention de départ, Intention à atteindre,* ***Stratégie pour atteindre l'intention***>. La première intention *Définir les points de vue* (cf. figure 1) est décrite par deux sections, correspondant aux deux stratégies mises en œuvre (parties 3.3.1, 3.3.2). Les deux intentions suivantes *Concevoir un modèle par point de vue* (partie 3.3.3) et *Consolider les modèles par point de vue* (partie 3.3.4) font chacune l'objet d'une section, correspondant à la stratégie mise en œuvre.

La méthode a été expérimentée et évaluée dans le cadre d'un projet pour le service de Formation Continue (FCV2A) de l'Université au mois de juin dernier. Ce projet de conception d'une application de gestion des présences pour la FCV2A nous permet aujourd'hui d'instancier la méthode avec un cas réel. Chaque section est donc illustrée par des éléments provenant de ce cas d'étude.

3.3.1. L'expression de visions pour définir les points de vue

La figure 3 présente la section <*Définir les points de vue, Définir les points de vue,* ***Par expression de visions***>. L'objet de cette section est de faire émerger les premiers points de vue des utilisateurs durant un atelier participatif nommé « atelier de cadrage ».

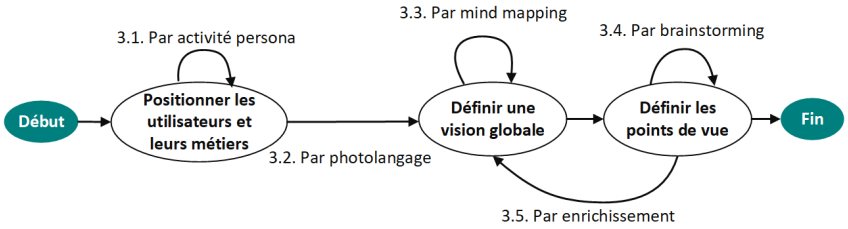


Figure 3. Section <Définir les points de vue, Définir les points de vue, ***Par expression des visions****>*

Durant cette étape, chaque utilisateur - au travers d'un persona - se positionne et positionne son métier (stratégie 3.1). Puis une activité de photolangage permet à chacun d'exprimer sa vision du domaine métier (stratégie 3.2) : nous décrivons cette activité en détail plus loin. Un partage sous la forme d'un mind mapping permet ensuite de constituer une vision globale du domaine à partir de toutes les visions individuelles (stratégie 3.3). Des discussions autour des éléments composant le brown paper permettent ensuite de dresser une première liste de points de vue (stratégie 3.4).

Le photolangage est l'activité centrale de cette première section pour la définition des points de vue. Cette activité se déroule en 5 étapes. Dans un premier temps, 100 photos sont disposées sur une table autour de laquelle les participants peuvent tourner. Chacune photo a un pouvoir symbolique fort : une longue route « sans fin », un site en construction, une équipe de sport, etc. Dans un second temps,

une question est posée à tous les participants. Dans le cadre de notre expérimentation avec le FCV2A, la question était « quelle est votre vision de la Formation Continue ? ». Durant cette étape, les participants ont 5 minutes pour choisir en silence une photo qui les aide à répondre à la question, et s'en emparer. Dans un troisième temps, les participants répondent à tour de rôle à la question posée en s'appuyant sur la photo choisie. Ils ont deux minutes pour parler, et peuvent intégrer la photo à leur discours comme ils le souhaitent. Le modérateur note alors les termes clés employés sur des post-its qu'il fixe au brown paper. A la fin du tour de table, les participants peuvent compléter le brown paper si des éléments leur semblent manquer. L'activité de photolangage présente de nombreux avantages pour ce type de stratégie (Bessell et al., 2007). D'une part, le photolangage est un outil facilitateur pour la prise de parole. En effet, dans le contexte d'un atelier participatif, les participants n'ont pas tous le même niveau d'appréhension face à l'expression orale en public, et certains peuvent avoir besoin d'aide. De plus, le choix d'une photo permet d'exprimer un point de vue individuel, sans exhaustivité, et de relativiser sa propre position. Cela permet de sortir des discours généralisant pour arriver à des prises de parole plus intéressantes. D'autre part, la conceptualisation ou la description abstraite d'un concept (ici une vision par exemple) est facilitée par l'usage du photolangage. En effet, il est parfois compliqué pour les participants de s'exprimer sur des sujets qu'ils abordent peu : manque d'idées, peur de dire des erreurs, etc. Le choix d'une photo peut être une source d'inspiration dans ces cas-là. De plus, le choix d'une photo étant totalement subjectif et personnel, il permet de reconnaître la caractère subjectif et partiel de ce que l'on exprime : on ne cherche pas à donner « la bonne réponse » mais on exprime un point de vue singulier.

3.3.2. L'élicitation des missions pour définir les points de vue

La figure 4 présente la section <*Définir les points de vue, Définir les points de vue,* ***Par élicitation des missions***>. L'objectif de cette section est d'identifier les missions par métier ainsi que les processus de travail afin d'affiner la liste de points de vue précédemment obtenue.

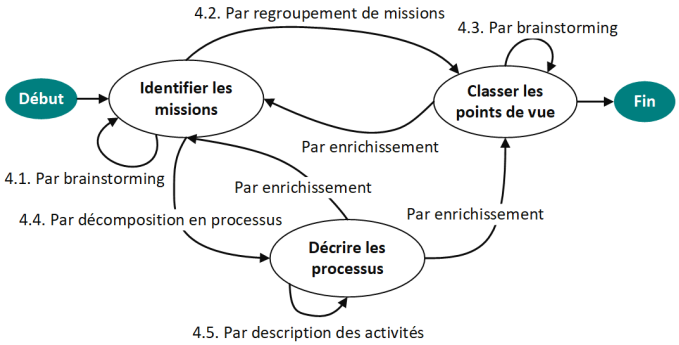


Figure 4. Section <Définir les points de vue, Définir les points de vue, ***Par élicitation des missions****>*

Cette deuxième partie de l'atelier de cadrage consiste à identifier les missions assignées à chacun des métiers et à les regrouper afin de les disposer sous les points de vue (stratégies 4.1 et 4.2). Une décomposition des missions en processus permet d'enrichir cette classification (stratégie 4.4) et d'obtenir une description précise de toutes les activités qui doivent être couvertes dans le cadre du projet (stratégie 4.5). La figure 5 présente une instanciation d'une partie du métamodèle de DEMOS correspondant à l'expérimentation avec le FCV2A. Deux points de vue ont émergé à la suite des discussions : le point de vue « gestion » et le point de vue « enseignement ».

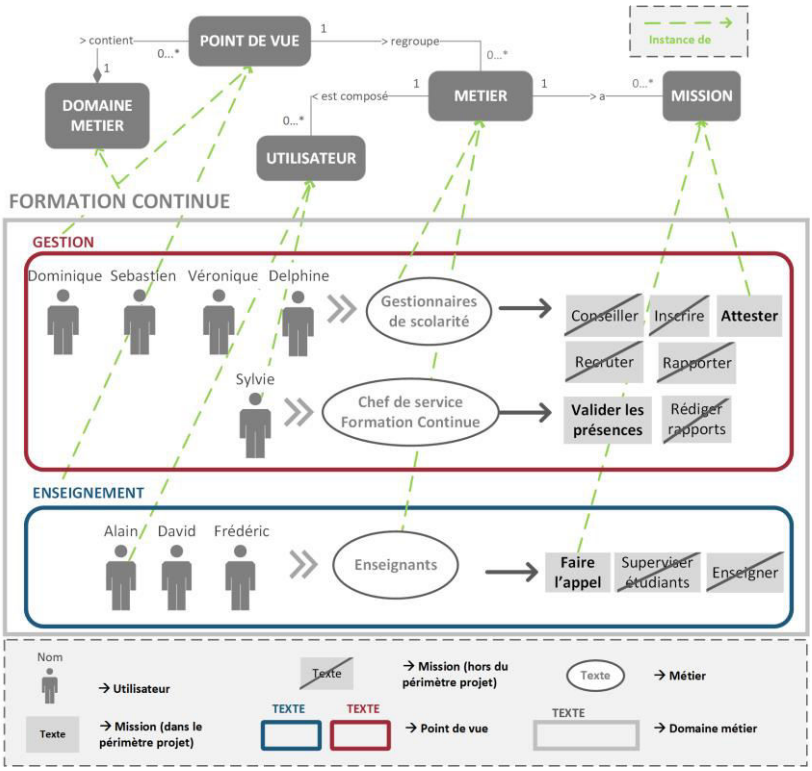


Figure 5. Première partie du métamodèle et son instanciation

3.3.3. La représentation conceptuelle pour concevoir les modèles par point de vue

La figure 6 présente la section <*Concevoir un modèle par point de vue, Concevoir un modèle par point de vue,* ***Par représentation conceptuelle***>. L'objectif de cette section est d'obtenir un modèle conceptuel par point de vue, c'est-à-dire de faire émerger les concepts, descripteurs et liens du domaine métier (voir le métamodèle de la méthode présentée figure 2).

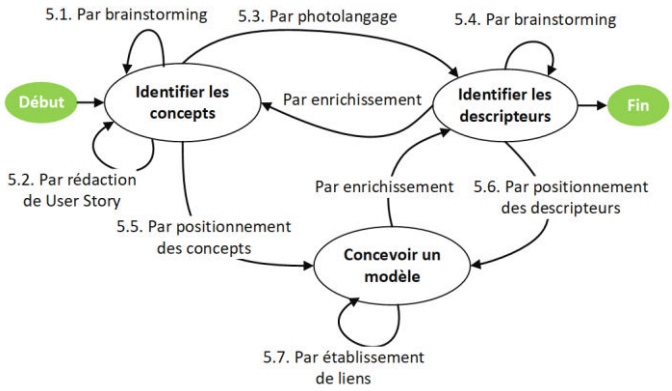


Figure 6. Section <Concevoir un modèle conceptuel par point de vue, Concevoir un modèle conceptuel par point de vue, ***Par représentation conceptuelle****>*

Chaque modèle conceptuel est réalisé lors d'un atelier « point de vue » qui ne regroupe que les utilisateurs concernés. Les artefacts les plus simples du diagramme UML sont utilisés. Dans un premier temps, les utilisateurs rédigent les User Story du projet (stratégie 5.2), et font ainsi émerger les premiers concepts (stratégie 5.1). Une activité de photolangage durant laquelle ils décrivent les concepts permet de dresser une liste de descripteurs (stratégie 5.3 et 5.4). Ils enrichissent ensuite les concepts et leurs descripteurs avec des liens (stratégie 5.7). Dans le cadre de l'expérimentation avec le FCV2A, deux ateliers « point de vue » ont donc eu lieu. Nous ne présentons dans cet article qu'une partie de chacun des modèles conceptuels obtenus (figure 7). Le vocabulaire décrivant le domaine métier est noté sous la forme de concepts et de descripteurs est propre à chacun des points de vue. Ici, par exemple, la notion d'étudiant/stagiaire est représentée sous des formes différentes. Des informations différentes intéressent chacun des points de vue, qui utilisent donc des descripteurs différents. Du point de vue « gestion », de nombreuses activités sont administrées, parmi lesquelles les cours (mais ils gèrent également les périodes de stage, le « travail libre », etc.). Du point de vue « enseignant », seuls les cours sont pris en compte.

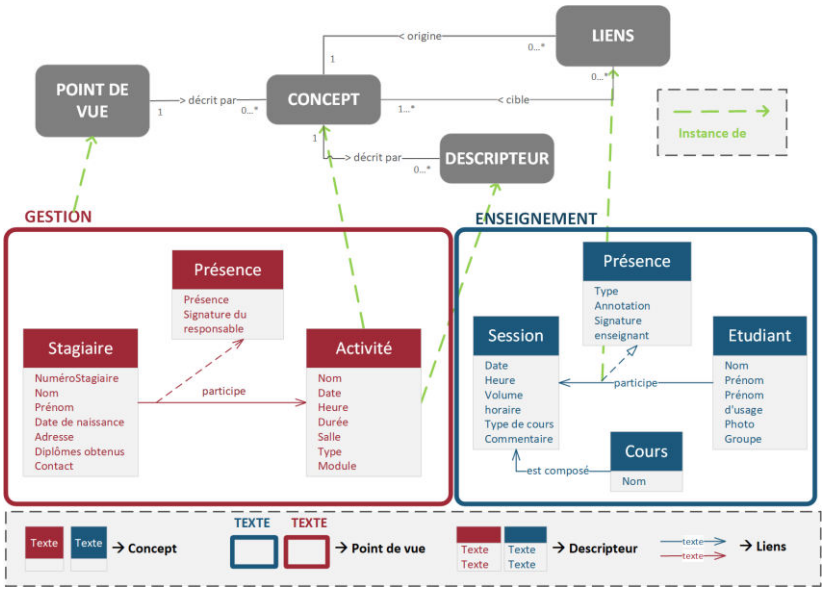


Figure 7. 2ème partie du métamodèle et son instanciation

3.3.4. La recherche de similarités pour consolider les modèles par point de vue

La figure 8 présente la section <*Consolider les modèles par point de vue, Consolider les modèles par point de vue,* ***Par recherche de similarités***>. L'objectif de cette section est de trouver les similarités entre les modèles conceptuels par point de vue, et par extension de mettre en lumière les points communs aux différents points de vue.

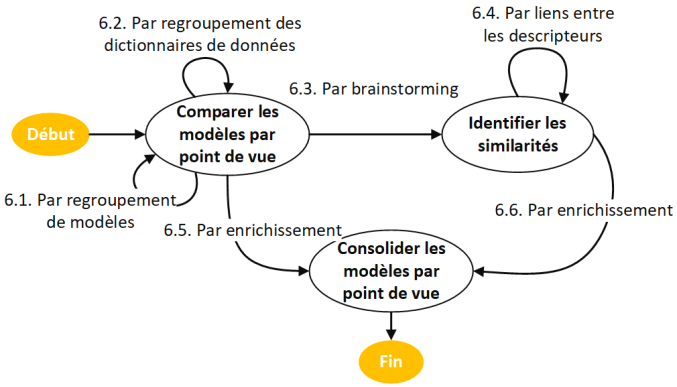


Figure 8. Section <Consolider les modèles par point de vue, Consolider les modèles par point de vue, ***Par recherche de similarités****>*

Pour cet atelier de mise en commun, le modérateur de la méthode prépare les dictionnaires des données de chacun des modèles et les affiche sur un brown paper (stratégie 6.1 et 6.2). Les participants maintenant réunis explicitent chacun des descripteurs, et les relient lorsqu'ils ont la même sémantique dans les deux modèles par point de vue (stratégie 6.3 et 6.4). Le résultat de cette étape est une liste de liens de similarités, qui permettront de consolider les modèles lors de l'implantation de ceux-ci. Durant l'expérimentation avec le FCV2A, les utilisateurs ont profité de cet atelier pour débattre des différentes façons de représenter leur domaine métier. Des liens de similarité ont été trouvés entre plusieurs descripteurs sur la partie des modèles conceptuels présentés à l'étape précédente. La figure 9 présente ces liens de similarité.

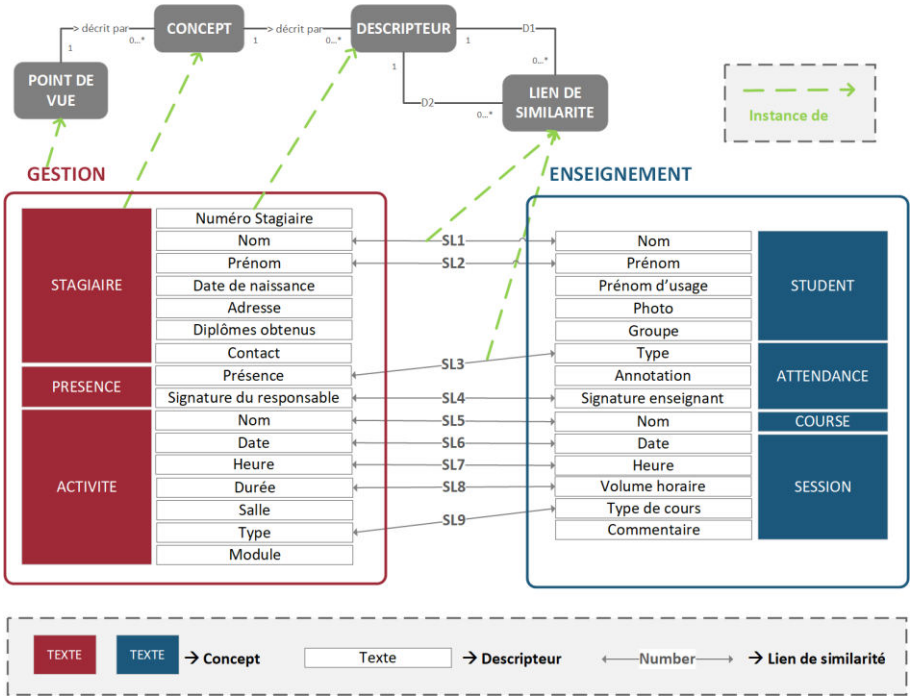


Figure 9. Troisième partie du métamodèle et son instanciation

4. Evaluation de la méthode

Notre ambition avec DEMOS est qu'une méthode structurée de conception logicielle puisse permettre l'intégration de démocratie dans le système d'information. Nous avons illustré notre proposition avec un cas concret provenant d'une expérimentation avec le FCV2A. A la suite de cette expérimentation de terrain, nous avons évalué la méthode au travers d'entretiens semi-structurés avec chacun des participants, en nous appuyant sur le protocole d'évaluation décrit dans le tableau 2.

Hypothèse
DEMOS permet de concevoir un système d'information de façon démocratique
Revue de la méthode (intentions et stratégies) et des résultats obtenus durant l'expérimentation ~ 15mn
Evaluation ~ 20 mn par participant
Analyse des intentions de la méthode
Les participants ont-ils compris les intentions de la méthode ?
Les participants ont-ils trouvé l'enchaînement des étapes de la méthode pertinent ?
Analyse des stratégies de la méthode
Les participants considèrent-ils que les stratégies ont permis d'atteindre les intentions ?
Les participants ont-ils trouvé les outils et techniques utilisés pertinents pour chaque stratégie ?
Analyse des résultats de la méthode
Les résultats obtenus sont ils en accord avec les intentions de la méthode

Tableau 2. Protocole d'évaluation de DEMOS

Les évaluations ont révélé que les participants ont compris les intentions de la méthode. Ils en ont saisi les deux composantes : un processus démocratique, pour la conception d'un système d'information démocratique qui respecte leurs points de vue. Bien que la notion de point de vue n'ait pas été simple à appréhender au départ, les participants ont peu à peu clarifié sa définition. Les participants ont trouvé l'ordonnancement des séquences de la méthode cohérent par rapport aux intentions fixées et ils estiment avoir été aidés par les outils et techniques employés, notamment par le jeu de photolangage. Au final, ils sont satisfaits des résultats de la méthode, qui correspond à ce qu'ils ont exprimés. Aujourd'hui, ils ne peuvent pas évaluer le logiciel, car il est en cours de développement.

5. Discussion et conclusion

La problématique adressée dans cet article est celle de la démocratie des systèmes d'information. Nous avons considéré ce sujet selon deux perspectives : comment intégrer la démocratie dans les démarches de conception de systèmes d'information et comment intégrer de la démocratie au sein du système d'information lui-même ? Le point de départ des discussions autour de la démocratie des systèmes d'information est la possibilité de débat autour des normes et valeurs encapsulées dans le système. Ce débat, s'il a lieu, facilite l'émergence de points de vue qui doivent ensuite être respectés et pris en compte, depuis la phase de recueil des exigences jusqu'à l'implémentation. Cela nécessite une approche participative impliquant les utilisateurs, un respect des différents points de vue composant le « groupe » utilisateurs et une traçabilité de ces mêmes points de vue. Durant l'expérimentation, et selon les résultats de l'évaluation de celle-ci, chacun des points mentionnés a été respecté. A l'avenir, nous souhaiterions explorer d'avantage le champ du développement, et notamment du développement agile pour prolonger l'intégration de démocratie jusqu'à cette étape. Une méthode telle que XP pourrait être une véritable extension de notre méthode de conception. Cependant, si DEMOS permet d'obtenir un système d'information démocratique du point de vue des utilisateurs, comment s'assurer que le processus de développement soit réellement

démocratique du point de vue des développeurs dans un projet XP (Jeffries, 2018) ? Il faudrait pour cela qu'il respecte un certain nombre de critères, à minima : participation de tous les développeurs aux prises de décision et respect des points de vue de chacun.

Remerciements

L'expérimentation illustrant cet article a été menée grâce au concours des gestionnaires de scolarité de la Formation Continue, de leur cheffe de service et de plusieurs enseignants de l'UT1 Capitole. Je tiens à les remercier pour le temps qu'ils ont accordé à ce projet, et pour leurs retours constructifs. Je remercie également Agnès Front et Dominique Rieu pour leur accueil au LIG Grenoble, et pour leur aide précieuse concernant la formalisation des méthodes de conception.

Bibliographie

Andre, K., and Christian, N. (2016). Participatory Design, User Involvement and Health IT Evaluation. Stud. Health Technol. Inform. 139–151.

Bessell, A.G., Deese, W.B., and Medina, A.L. (2007). Photolanguage: How a Picture Can Inspire a Thousand Words. Am. J. Eval. *28*, 558–569.

Brey, P.A.E. (2010). Values in Technology and Disclosive Computer Ethics. In The Cambridge Handbook of Information and Computer Ethics., L. Floridi, ed. (Cambridge University Press), pp. 41–58.

Chamberlain, S., Sharp, H., and Maiden, N. (2006). Towards a Framework for Integrating Agile Development and User-Centred Design. In Extreme Programming and Agile Processes in Software Engineering, P. Abrahamsson, M. Marchesi, and G. Succi, eds. (Berlin, Heidelberg: Springer Berlin Heidelberg), pp. 143–153.

Damodaran, L. (1996). User involvement in the systems design process-a practical guide for users. Behav. Inf. Technol. *15*, 363–377.

Darses, F. (2004). Chapitre premier. La conception participative : vers une théorie de la conception centrée sur l'établissement d'une intelligence mutuelle. In Le Consommateur Au Coeur de L'innovation, (Paris: CNRS Editions), p.

Darses, F., Détienne, F., and Willemien, V. (2001). Assister la conception : perspectives pour la psychologie cognitive ergonomique. (Nantes: Epique), p.

Dell'Era, C., and Landoni, P. (2014). Living Lab: A Methodology between User-Centred Design and Participatory Design: Living Lab. Creat. Innov. Manag. *23*, 137–154.

Ferrario, M.A., Simm, W., Newman, P., Forshaw, S., and Whittle, J. (2014). Software engineering for "social good": integrating action research, participatory design, and agile development. In Companion Proceedings of the 36th International Conference on Software Engineering - ICSE Companion 2014, (Hyderabad, India: ACM Press), pp. 520–523.

Floridi, L. (2010). The Cambridge handbook of information and computer ethics (Cambridge; New York: Cambridge University Press).

Greenbaum, J.M. (1991). Design at work: cooperative design of computer systems (Hillsdale, N.J: L. Erlbaum Associates).

Highsmith, J., and Cockburn, A. (2001). Agile software development: the business of innovation. Computer *34*, 120–127.

van den Hoven, J. (2008). Moral Methodology and Information Technology. In The Handbook of Information and Computer Ethics, K.E. Himma, and H.T. Tavani, eds. (Hoboken, NJ, USA: John Wiley & Sons, Inc.), pp. 49–67.

Jeffries, R. (2018). Developers Should Abandon Agile.

Kautz, K. (2011). Investigating the design process: participatory design in agile software development. Inf. Technol. People *24*, 217–235.

Kensing, F., and Blomberg, J. (1998). Participatory Design: Issues and Concerns. Comput. Support. Coop. Work CSCW *7*, 167–185.

Kujala, S. (2003). User involvement: A review of the benefits and challenges. Behav. Inf. Technol. *22*, 1–16.

Mahaux, M., Nguyen, L., Gotel, O., Mich, L., Mavin, A., and Schmid, K. (2013). Collaborative creativity in requirements engineering: Analysis and practical advice. (IEEE), pp. 1–10.

McConnell, S. (2006). Rapid development: taming wild software schedules (Redmond, Wash: Microsoft Press).

Millington, D., and Stapleton, J. (1995). Developing a RAD standard. IEEE Softw. *12*, 54–55.

Mingers, J., and Walsham, G. (2010). Toward Ethical Information Systems: The Contribution of Discourse Ethics. MIS Q *34*, 833–854.

Rolland, C., Prakash, N., and Benjamen, A. (1999). A Multi-Model View of Process Modelling. Requir. Eng. *4*, 169–187.

Salles, M. (2015a). Responsabilité économique et sociale des concepteurs de systèmes d'information : contribution à une éthique appliquée. Innovations *46*, 197.

Salles, M. (2015b). Decision-making and the information system. (John Wiley & Sons).

Salles, M., and Colletis, G. (2008). How to deal with the conflicting views of the world expressed in regional economic development policies? In International Conference of Territorial Intelligence, Besançon 2008., (Besançon, France), p. 10.

Sanders, E.B.-N., and Stappers, P.J. (2008). Co-creation and the new landscapes of design. CoDesign *4*, 5–18.

Schwartz, L., Vergnol, L., Gronier, G., Vagner, A., Altenburger, T., and Battisti, S. (2009). Comment concilier agilité et conception centrée utilisateurs dans un projet de développement? (ACM Press), p. 337.

Sharp, H., and Robinson, H. Integrating user-centred design and software engineering : a role for extreme programming ?

Simonsen, J. (2013). Routledge international handbook of participatory design (London: Routledge).

Walsham, G. (2012). Are we making a better world with ICTs? Reflections on a future agenda for the IS field. J. Inf. Technol. *27*, 87–93.

Éco-conception logicielle pour systèmes durables : retours d'expérience en matière de mobilité et de gestion forestière

Christophe Ponsard [1], **Bérengère Nihoul** [1], **Mounir Touzani** [2]

1. *CETIC - Centre de recherche, Gosselies, Belgique*
   {christophe.ponsard,berengere.nihoul}@cetic.be
2. *Chercheur indépendant, Toulouse, France*
   mtouzani64@gmail.com

RÉSUMÉ. *La durabilité est une préoccupation majeure du 21ème siècle. La révolution numérique engendre des opportunités mais également de nouvelles menaces. Le constat actuel est que les décisions de conception liées à la durabilité restent souvent implicites et résultent d'un processus impliquant des synergies et des compromis entre de nombreuses exigences non-fonctionnelles. L'objectif de cet article est d'identifier ces décisions et de mieux expliciter ce processus pour s'assurer de sa cohérence. Notre démarche s'appuie une série d'outils méthodologiques identifiés dans la littérature couplée avec deux retours d'expérience : un partage de service de navettes, analysé de manière rétrospective, et un système de gestion éco-responsable d'un domaine forestier, mené selon une démarche participative.*

ABSTRACT. *Sustainability has become a major concern of our time. The present digital revolution is the source of both opportunities and threats. A fact is that sustainability design decisions remain most of the time implicit and result from a process involving synergies and trade-offs among many non-functional requirements. The purpose of this paper is to identify such decisions and make the process more explicit and tool supported. For this, we rely on two case studies (1) in mobility, to report on analysis and design tools, and (2) in forest management, to validate participatory tools.*

MOTS-CLÉS : *Informatique durable, développement durable, living lab, analyse orientée but, optimisation, étude de cas*

KEYWORDS: *Sustainable Computing, Sustainable Development, Living Lab, Goal-Oriented Analysis, Optimisation, Case Study*

---

1. Introduction

La durabilité est devenue un défi majeur de notre siècle. Nous sommes actuellement confrontés à la réalité et même aux conséquences de l'atteinte des limites fondamentales des ressources de notre planète Terre. En 1987, le rapport Brundtland des Nations Unies avait déjà défini le développement durable (ou soutenable) comme « un développement qui répond aux besoins du présent sans compromettre la capacité des générations futures à répondre aux leurs » (Nations, 1987). Depuis ce rapport, l'énorme expansion des technologies de l'information et de la communication (TIC) a créé une nouvelle économie digitale qui constitue d'une part une opportunité pour une gestion plus durable de nos systèmes notamment via les possibilités de dématérialisation et d'optimisation de l'usage de nos ressources. D'autre part, ce progrès technologique peut également générer des menaces, compte tenu de l'énorme consommation de ressources pour faire fonctionner les terminaux, les réseaux, les centres de stockage et plus récemment les cryptomonnaies ainsi que la technologie blockchain (Vickery, 2012). En matière de consommation énergétique, la part de notre nouvelle « économie digitale » a bondi de 1% en 2006, 2% en 2012 et jusqu'à 7% en 2018. De nombreux experts estiment que l'économie digitale pourrait approcher les 20% vers 2020, avec une croissance de l'ordre de 10% par an (Greenpeace, 2017).

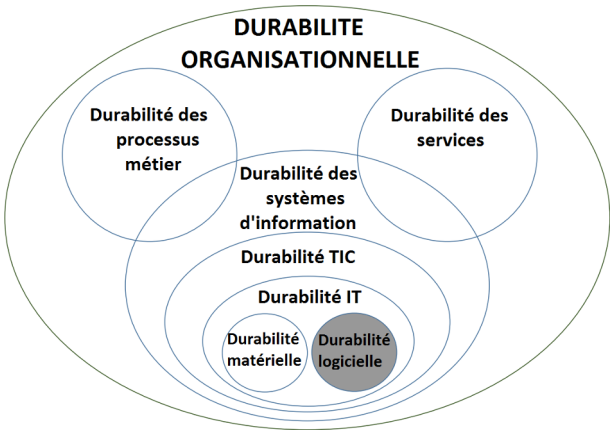


Figure 1. Différents niveaux de durabilité (Calero, Piattini, 2015)

Le développement durable repose sur trois grands piliers (IUCN, 2006) :

– une *dimension environnementale* visant à préserver, améliorer et valoriser l'environnement et les ressources naturelles sur le long terme,

– une *dimension sociale* afin de satisfaire les besoins humains de manière équitable, notamment en matière de santé, logement, consommation, éducation, etc. ,

– une *dimension économique* pour développer la croissance et l'efficacité économique, à travers des modes de production et de consommation durable.

Au sein d'une organisation, l'aspect logiciel durable est enfoui au coeur de plusieurs couches successives depuis la couche métier traitant des aspects processus, des

services et du système d'information (SI), puis de sa composante TIC avant d'atteindre le niveau IT d'infrastructure matérielle et logicielle, comme illustré à la figure 1 tirée de (Calero, Piattini, 2015). Malgré ce manque de visibilité, les logiciels jouent un rôle important et leur durabilité a été identifiée parmi les principales exigences non fonctionnelles du XXIe siècle (Penzenstadler *et al.*, 2014).

Cependant, la conception de logiciels « durables » n'est pas une tâche facile, car différents aspects liés aux différentes facettes de la durabilité, doivent être pris en compte. Dans cet article, nous nous appuyons sur la terminologie définie par (Calero, Piattini, 2015), qui caractérise deux dimensions complémentaires : la durabilité « PAR » les SI/TIC (aussi appelée « ICT4Green ») et la durabilité « DANS » les SI/TIC (aussi appelé « Green-IT »). Dans le cadre de cet article, nous nous intéressons aux dimensions plus spécialisées suivantes, touchant au logiciel et à l'infrastructure, tout en gardant le système global en perspective.

– *La durabilité PAR le logiciel :* comment le logiciel aide-t-il à assurer la durabilité du système dans lequel il est utilisé ?

– *La durabilité DANS le logiciel :* dans quelle mesure le logiciel est-il durable, du point de vue de l'ingénierie, de l'exécution et de la maintenance du logiciel ?

– *La durabilité DANS l'IT :* dans quelle mesure l'infrastructure informatique est-elle durable (principalement en termes d'efficacité énergétique) ?

Cet article s'appuie sur le processus suivant, qui implique deux études de cas entrecoupées par une phase de prise de recul :

– *un premier cas d'étude en mobilité* très riche au niveau des aspects durabilité. Il a été mené sans démarche explicite et analysé en aval du développement afin d'identifier comment ces dimensions avaient été traitées et si certains aspects auraient pu être traités différemment par une démarche plus consciente

– *une phase de recul*, où des outils et des méthodes intéressantes ont été documentés plus en détail et croisés avec la littérature disponible

– *un second cas d'étude en matière de gestion durable dans le domaine forestier*, où des outils sont appliqués en amont, de manière consciente

Cet article est structuré de la manière suivante : la section 2 présente une série d'outils et méthodes pertinentes en se référant à la littérature du domaine, basé également sur le premier cas d'étude a posteriori qui fait l'objet de la section 3. Ensuite, la section 4 propose un second cas d'étude forestier réalisé plus en amont et qui met l'accent sur les outils participatifs. Enfin, nos conclusions et perspectives sont présentées à la section 5.

2. Outils méthodologiques de conception de systèmes durables

Cette section étudie trois types d'outils utiles pour la conception de système durables : les méthodes dirigées par les objectifs, les méthodes participatives et les bonnes pratiques d'éco-conception.

2.1. Analyse par objectifs

L'ingénierie des exigences orientée buts propose des notations permettant de structurer des buts de différents types d'agents à différents niveaux de raffinements (du plus stratégique au plus opérationnel) et de raisonner également sur la manière dont il peuvent se renforcer ou à l'inverse entrer en conflit. Chaque notation dispose d'une spécificité. La méthode i* (Yu, Mylopoulos, 1997) est plus centrée sur l'organisation, tandis que KAOS (Lamsweerde, 2009) se concentre sur la partition entre les agents du système (dont le logiciel) et de l'environnement (dont les utilisateurs). Enfin, URN-GRL (International Telecommunication Union, 2012) est une variante standardisée orientée entreprise. Concernant la durabilité, différentes techniques ont été proposées et sont rapidement évoquées dans cette section.

Le manifeste de Karlskrona contient neuf principes relatifs à l'aspect systémique, multidisciplinaire, multi-niveaux et le long terme (Becker *et al.*, 2015). Une série de bonnes pratiques dérivée de ce manifeste, a également été proposée pour guider tout le cycle (Oyedeji, Penzenstadler, 2018).

Les trois dimensions de la durabilité présentées dans l'introduction peuvent être visualisées à l'aide d'un diagramme de Venn. Des vues plus riches peuvent aussi être utilisées, notamment le diagramme en toile d'araignée qui capture plus de dimensions et les mesure par rapport à une cible (Becker *et al.*, 2016).

Des structurations par arbres de buts ont été proposées à la fois pour les notations KAOS (Mahaux *et al.*, 2011) et i* (Cabot *et al.*, 2009). Au delà de l'étape d'élaboration du modèle de buts, une guidance pour la prise de décision est également disponible et proposée pour KAOS (Stefan *et al.*, 2011). Il s'agit d'estimer des valeurs du modèle et les coûts/bénéfices/risques associés. Les buts doivent être mesurables, donc disposer d'indicateurs mesurables ou KPI (Key Performance Indicator) dans une échelle pertinente et non pas dans une unité abstraite ou uniquement monétaire. Le besoin de confronter des indicateurs de natures différentes est un moteur qui permet de mettre en place des discussions de nature transverse et multidisciplinaire.

Des techniques de pensée systémique (« system thinking ») peuvent être utilisées pour approfondir l'analyse des interactions positives ou négatives entre des buts, notamment via l'identification de boucles de renforcement et d'affaiblissement (Williams *et al.*, 2017), voire les simuler au moyen de modèles de dynamique des systèmes.

2.2. Approche participative et Living Labs

Un Living Lab est un laboratoire d'innovation ouvert, plaçant les usagers au cœur de la création, afin de concevoir et de développer des produits/services innovants pour répondre aux attentes et besoins de la société (Kusiak, 2007). Il implique des collectivités locales, des entreprises, des laboratoires de recherche, ainsi que des utilisateurs potentiels. Il ajoute ainsi une composante citoyenne à des modèles d'innovation de type triple hélice, associant université, industrie et gouvernement. Un Living Lab est

généralement ancré dans un domaine spécifique et repose en partie sur sa capacité à créer et animer une communauté d'usagers et d'experts.

Le développement logiciel et durable est particulièrement pertinent à analyser dans une telle perspective, au vu de la présence généralisée du logiciel et ses interactions fortes avec les dimensions durables, dans un contexte de la prise de conscience citoyenne face à ses enjeux.

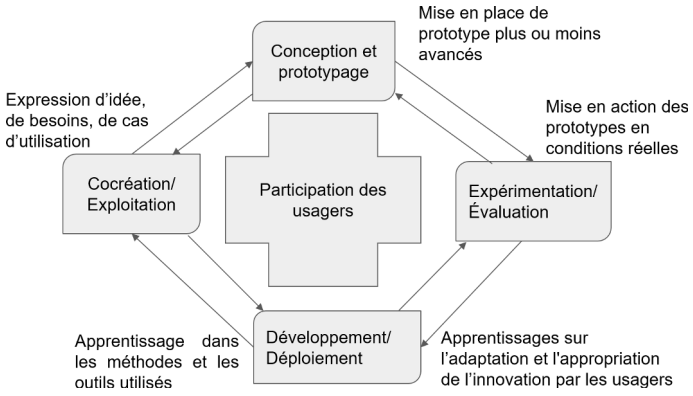


Figure 2. Principe de fonctionnement d'un Living Lab

La méthodologie Living Lab comprend quatre phases illustrées à la figure 2 :

1. l'idéation via des techniques de co-création en atelier avec des usagers,
2. la conception par prototypage, généralement en plusieurs étapes intermédiaires avec des retours vers une phase de co-création,
3. l'expérimentation et la validation qui, après plusieurs retours possibles vers des phases de co-création et de conception, aboutissent à une phase de déploiement,
4. ce déploiement apporte des enseignements à un nouveau cycle de co-création en termes d'usages, d'évolution des produits/services et de méthodologie.

Des Living Lab ont déjà abordé le contexte durable. Ainsi, la transition urbaine durable a été étudiée dans plusieurs pays européens et a montré des impacts directs, indirects et diffus sur les dimensions économiques et environnementales, bien que difficiles à mesurer précisément (Schliwa *et al.*, 2015). Les villes intelligentes intègrent aussi souvent un tel outil, notamment en Finlande où une application de gestion des flux de navetteurs a été expérimentée dans ce contexte (Alam, Porras, 2018). Des facteurs clés de succès concernent l'implication d'acteurs de la « quadruple hélice » de l'innovation, c'est-à-dire que la composante citoyenne doit aussi être couplée à des acteurs issus de l'université, de l'industrie et du gouvernement. La phase d'identification des défis est cruciale et s'appuie naturellement sur les outils d'analyse par objectifs. Le prototype doit soutenir en priorité la validation des questions et non pas sa qualité (notamment il peut être inefficace s'il est utilisé à petite échelle). Une démarche de type Agile est recommandée pour ce type de développement (U4IOT, 2018).

2.3. Méthodes d'éco-conception logicielle

L'éco-conception, dans son sens large est un terme désignant la volonté de concevoir des produits respectant les principes du développement durable et de l'environnement, en recourant « aussi peu que possible aux ressources non renouvelables en leur préférant l'utilisation de ressources renouvelables, exploitées en respectant leur taux de renouvellement et associées à une valorisation des déchets qui favorise le ré-emploi, la réparation et le recyclage » (Ademe, 2012).

En matière de conception logicielle, ceci couvre d'une part une dimension « produit »: ce que le logiciel mobilisera comme ressources de calcul, stockage et réseau lors de son exécution. D'autre part, il y a une dimension « processus » relative aux ressources, en particulier humaines, impliquées dans tout le cycle de vie, depuis les exigences jusqu'à la maintenance (Penzenstadler, 2013).

Au niveau du produit logiciel durable, le travail relève de la formation technique en conception et en déploiement de logiciels. Ces pratiques sont souvent alignées à la recherche de performance mais vont plus loin dans le sens où on peut délibérément simplifier ou éliminer des fonctionnalités en comparant la valeur générée par rapport au coût en ressources. De manière globale, 64% des fonctionnalités livrées sont rarement ou non utilisées (Johnson, 2002). Pour soutenir la démarche, des références existent dans divers domaines. Par exemple, pour les applications web, un guide de 115 conseils a été proposé par un collectif et édité sous une licence créative commons (Bordage, 2015). Concernant le déploiement, les possibilités offertes par les infrastructures virtualisées, sous forme de conteneurs ou de microservices, conduisent à des stratégies durables si elles sont configurées adéquatement (Djemame *et al.*, 2014).

Au niveau du processus de développement, les bonnes pratiques de développement sont d'application. L'attention portée aux exigences par une démarche d'analyse des objectifs et une implication participative permet de limiter le risque que le logiciel ne réponde pas aux besoins, donc d'allonger le cycle de développement. Les méthodes agiles se prêtent naturellement aussi bien à une telle démarche (Tate, 2005).

3. Analyse a posteriori - cas d'étude en mobilité

3.1. Description du cas d'étude et de son contexte

SAM-Drive (ou simplement « SAM » en abrégé) est un service de navettes collectives avec un chauffeur opérant dans la banlieue de Bruxelles en Belgique (Michel Renders, 2012). La mobilité est un défi majeur en matière de développement durable (Banister, 2008). C'est un problème particulièrement marqué dans cette région à forte densité de population avec une offre de transport en commun ne couvrant pas tous les besoins. L'offre SAM lui est complémentaire pour des transports à la demande, visant des publics diversifiés (adolescents, écoliers, sportifs, personnes âgées, personnes isolées, etc.), des créneaux horaires extrêmement flexibles, des besoins spécifiques (re-

tours de soirée, aéroports, gares, médecins, hôpitaux, activités parascolaires...). Les objectifs prioritaires de SAM sont les suivants :

- hausse de la mobilité des citoyens, surtout pour les segments mal couverts
- réduction de l'empreinte environnementale
- renforcement de la sécurité routière
- fourniture d'un rapport qualité/prix très concurrentiel
- création d'emplois durables pour des chauffeurs souvent remis au travail
- contribution réelle aux différents plans publics régionaux de mobilité
- partage de l'expérience au niveau national et international

Un obstacle majeur à l'efficacité et la croissance de SAM était sa sous-utilisation du potentiel des TIC avec une gestion largement manuelle des demandes et du dispatching des courses vers les conducteurs, avec un support essentiellement suivi via des feuilles de calcul. Malgré la présence claire d'objectifs durables, la première priorité de SAM était la mise en place d'une plateforme IT permettant d'automatiser et optimiser ses processus essentiellement, afin de diminuer les coûts d'exploitation tout en minimisant le temps passé sur la route (y compris dans les embouteillages) et en minimisant les temps morts des conducteurs. Indirectement, ceci permet de réduire les émissions de CO_2. La seconde priorité de SAM était d'augmenter le volume et la réactivité du traitement des demandes tout en réduisant le niveau de stress du personnel. Enfin, SAM désirait mettre en œuvre une tarification encourageant le partage via une dégressivité en fonction de l'allongement de la course.

Bien que les objectifs de développement durable soient largement identifiables, ils n'ont pas été identifiés ni traités de manière explicite durant le processus de conception de la plateforme logicielle, avec cependant une équipe de développement relativement bien sensibilisée à cette thématique.

3.2. Méthode d'analyse

Le but de l'analyse est de réaliser une rétrospective du projet afin de mettre en évidence dans quelle mesure des objectifs de durabilité ont été identifiés et traités en tout ou partie malgré la démarche implicite suivie. Il s'agit ensuite, sur cette base, d'illustrer l'intérêt de l'application d'outils et techniques décrits à la section 2.

Notre analyse se base sur les questions de recherche suivantes :

– RQ1 - Comment décider des objectifs de développement durable de manière cohérente au niveau du système ?

– RQ2 - Comment mettre en œuvre durablement le logiciel de manière cohérente PAR/DANS à différents niveaux du système/logiciel/infrastructure ?

– RQ3 - Comment évaluer la durabilité du système résultant ?

Dans cette section, divers outils seront aussi illustrés directement dans leur mise en œuvre. Ils seront présentés plus en détail dans la section 3.

3.3. RQ1 - Analyse des objectifs de développement durable

Afin d'analyser les objectifs de développement durable, les objectifs de SAM décrits à la section précédente ont été analysés. Deux outils se sont avérés utiles :

– une visualisation via un diagramme de Venn permet de positionner et vérifier la couverture des objectifs par rapport aux trois piliers du développement durable (environnemental, social et économique) (IUCN, 2006). La contribution à plusieurs axes est représentée via un placement au niveau des intersections des ensembles.

– une structuration globale des objectifs par arbres de buts selon des méthodologies d'ingénierie des exigences (ici avec la variante KAOS (Lamsweerde, 2009)). Ceci met en évidence quel composant logiciel supporte quel objectif durable et également les synergies et conflits éventuels entre les objectifs de développement durable et d'autres types d'objectifs fonctionnels et non-fonctionnels.

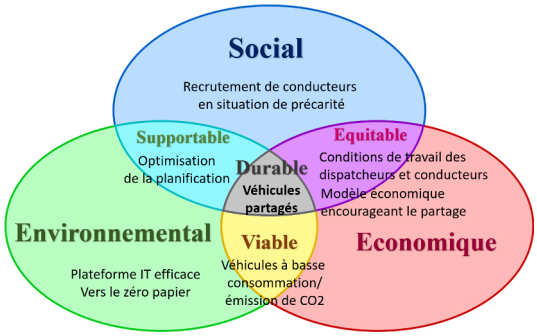


Figure 3. Objectifs de SAM selon les trois piliers du développement durable

La figure 3 présente le positionnement durable des objectifs identifiés :

– *Encourager le partage de courses via un modèle de tarification dégressif en fonction du détour*. Les trois dimensions sont abordées, car l'auto-partage réduit l'empreinte environnementale et l'utilisateur est économiquement récompensé. Sur le plan social, les voyages ensemble sur des trajets récurrents créent des liens.

– *Recruter des travailleurs en situation de précarité*, souvent issus du chômage et qui ont retrouvé un emploi sûr, contrairement aux emplois précaires proposés par la nouvelle économie comme Uber (Younglai, 2015). Cette dimension est sociale.

– *Utiliser une plateforme IT efficace*. Les demandes sont traitées efficacement avec un minimum de ressources informatiques et la suppression de la feuille de route papier, ce qui réduit l'empreinte environnementale.

– *Améliorer les conditions de travail*. La nouvelle solution informatique mieux conçue réduit le stress chez les dispatcheurs et les conducteurs. Elle augmente aussi les capacités opérationnelles. L'amélioration est donc à la fois économique et sociale.

– *Optimiser la planification* pour répondre aux contraintes des conducteurs et minimiser le temps sur la route dans les embouteillages, donc réduire les émissions de carbone. Cet objectif est à la fois environnemental et social.

– *Utilisation de véhicules basse consommation/émission*. Les véhicules sont thermiques à basse consommation et aux dernières normes en matière d'émission avec une évolution prévue vers de l'électrique. Cet objectif est environnemental et économique.

Dans le cas SAM, les objectifs durables étaient aisés à découvrir. Une démarche de décomposition dirigée par les objectifs est utile pour systématiser leur découverte mais aussi pour identifier comment ils s'alignent plus ou moins bien avec d'autres objectifs du système à mettre en place. La figure 4 illustre cette démarche avec une première décomposition réalisée selon les parties prenantes : externes (clients) pour la partie de gauche (objectif de partage) et internes (employés) pour la partie de droite (objectif de qualité de travail). Pour la partie du client, les objectifs plus fins sont relatifs au modèle de tarification, à la prise en compte des contraintes du client, ainsi que la précision et la fiabilité. Ils se traduisent en exigences sur des composants de facturation, de simulation des coûts et d'estimation du temps de trajet. La partie relative au personnel porte sur les dispatcheurs (automatisation/optimisation) et les conducteurs (qualité du planning). Dans la figure 4, les buts durabilité sont annotés avec la mention "DUR". Notons que les buts avec fond jaune (politique de recrutement et type de véhicule) ne sont pas sous la responsabilité d'un logiciel.

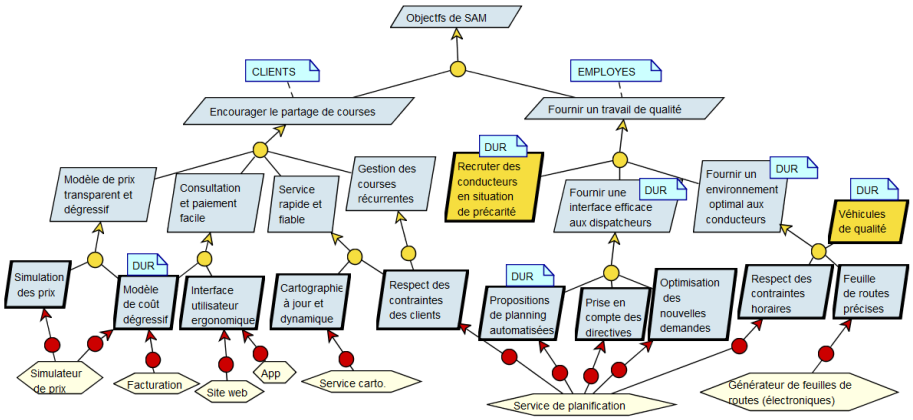


Figure 4. Arbre de raffinement des objectifs pour le système de navettes partagées

Une faiblesse identifiée a posteriori, est la perception limitée et indirecte que SAM avait des besoins des utilisateurs, essentiellement via des retours de conducteurs ou de dispatcheurs en cas de problème. Une implication plus étroite des utilisateurs est possible via des techniques plus récentes de co-création et le recours à des Living Labs présentés à la section 2. Leur utilisation permet de capturer des scénarios innovants de mobilité mais aussi d'initier d'emblée une meilleure adoption.

3.4. RQ2 - Mise en œuvre durable PAR/DANS le logiciel et l'infrastructure

Cette section illustre la durabilité DANS/PAR le logiciel sur deux éléments clefs de SAM : la technologie d'optimisation et l'architecture de l'application.

3.4.1. Moteur d'optimisation efficace et effectif

L'optimisation contribue aux deux facettes de la durabilité, à la fois via son efficacité en mobilité et via une consommation raisonnée de ressources informatiques.

Concernant l'efficacité, le service de planification gère de nombreuses contraintes relatives aux courses des clients et au temps de travail des conducteurs. Une approche durable établit aussi un compromis entre différents objectifs : il faut à la fois répondre rapidement à une demande, augmenter le taux de remplissage des véhicules mais aussi laisser le contrôle à l'équipe de dispatching tout en lui facilitant au maximum la tâche. Pour réaliser ce type de compromis, des KPI pour chaque type d'exigence peuvent être définis : on peut également mesurer le nombre de courses traitées par rapport à celles refusées, la distance totale parcourue, le remplissage moyen des véhicules, la charge de travail des conducteurs, etc.

Concernant l'efficience, une attention importante a été portée au choix de la technologie d'optimisation utilisée. Au lieu de considérer un moteur implémenté dans un langage proche de la machine tel que C/C++, le choix a été fait sur des langages supportant un style plus déclaratif et des constructions algorithmiques efficaces. Il s'agit du moteur Scala OscaR.CBLS, (OscaR, 2012) qui utilise la recherche locale basée sur des contraintes (Van Hentenryck, Michel, 2009). Cette approche est payante car le surcoût de mise en place d'un langage plus efficace est très rapidement amorti par la réduction de la complexité des problèmes de routage qui sont de nature combinatoire. Des confrontations à des outils C++ tels que Google OR-Tools ont confirmé ceci (Landtsheer *et al.*, 2018). Par ailleurs, la meilleure expressivité du langage permet de réduire fortement le temps de développement à qualité égale de la solution.

3.4.2. Gestion durable de l'infrastructure IT

L'interface utilisateur de dispatching s'appuie sur le moteur OscaR qui doit disposer d'une matrice des distances potentiellement pour toutes les paires de points impliqués dans un circuit. En plus de sa taille très importante, cette matrice peut idéalement renvoyer des fonctions de temps de trajet au lieu d'un nombre unique, car la durée du trajet peut varier au cours de la journée. Pour réduire le temps de calcul, des algorithmes efficaces basés sur les « hiérarchies de contraction » peuvent être mis en oeuvre (Geisberger *et al.*, 2008). En outre, un mécanisme de cache d'éviter de recalculer inutilement des temps déjà connus.

Au niveau informatique, l'architecture a été déployée sous forme de conteneurs basés sur la technologie Docker (Hykes, 2013). Du point de vue de l'utilisation des ressources, c'est beaucoup plus léger que d'utiliser un ensemble de machines virtuelles. C'est également plus facile à gérer à travers les différents environnements de développement, test et production (Nagler *et al.*, 2015).

3.5. RQ3 - Évaluation du niveau de durabilité

Comme indiqué précédemment, des KPI permettent le suivi non seulement d'objectifs opérationnels mais aussi d'objectifs de durabilité. Ces indicateurs peuvent être définis à différents niveaux de granularité dans l'arbre des buts : au niveau le plus bas, des indicateurs spécifiques sont utilisés, par exemple : le nombre et pourcentage de demandes traitées par les dispatcheurs, le nombre de kilomètres parcourus et son équivalent en CO_2 d'un point de vue durabilité. A un niveau plus global, des consolidations sous forme monétaires peuvent être réalisées selon le modèle de coût appliqué. Un tableau de bord permettant le suivi de ces KPI est illustré en partie à la figure 5.

PLANNING	… Nouvelles demandes à traiter	… Demandes en attente du client
CONVERSIONS	Ces 30 derniers jours … Nouveaux clients	Ces 30 derniers jours … Premières demandes
AUJOURD'HUI	… Passagers transportés	… € Chiffre d'affaire
CÔTÉ CHAUFFEURS	… Km Parcourus aujourd'hui	… H Heures payées

Figure 5. Quelques KPI métier du tableau de bord

Au niveau technique, les performances techniques sont également surveillées, en particulier pour la partie optimisation (par exemple, le cas de problèmes sur-contraints) et la partie matricielle (taux de réussite du cache et exécution du calcul de nouvelles valeurs). Un tableau de bord technique permet de consulter ces indicateurs et s'appuie sur le système de surveillance Open Source Promotheus (Promotheus, 2018). Ce composant est facile à configurer grâce à l'ajout aisé de connecteurs, dont beaucoup sont disponibles dans le commerce. Il peut collecter les données de manière efficace et fournit aussi de bons tableaux de bord qui se rapportent aux indicateurs techniques.

4. Cas d'étude a priori : gestion durable d'un domaine forestier

Cette section présente notre second cas d'étude réalisé en amont du développement. Cette étude est encore en phase d'analyse et se focalise essentiellement sur des outils complémentaires tels que les démarches d'innovation ouverte et participative.

4.1. Description

La forêt wallonne s'inscrit dans un bassin forestier homogène plus large, qui va de la Rhénanie au Luxembourg, en passant par la Lorraine. Ce territoire, appelé Grande Région, totalise 2.375.000 ha de forêt, dont 40% sont privés. Un problème majeur en forêt privée, est le fort taux d'abandon. Il est donc important de pouvoir localiser des zones où le reboisement est déficient pour pouvoir proposer aux propriétaires des

itinéraires de renouvellement innovant et moins onéreux. Par ailleurs, la forêt dispose d'une forte attractivité touristique (balades, observations, ressourcement, etc.) liée à son capital de nature et de biodiversité.

4.2. Objectifs durables

Les produits et services en place n'étant pas toujours cohérents, l'intégration dans le cadre environnemental des dimensions sociales de tourisme et économiques d'exploitation peut être revisitée. Pour des raisons de place, seul le positionnement général des objectifs durables est présenté à l'aide du diagramme de Venn à la figure 6.

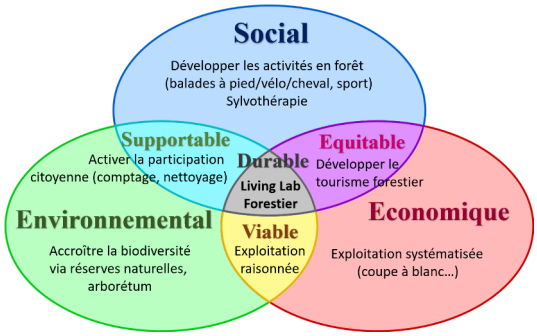


Figure 6. Positionnement des objectifs durables du cas forestier

Notons que le concept durable est largement intégré dans les forêts européennes avec une exploitation rationnelle des ressources, une planification des coupes et l'entretien des forêts, mais principalement avec une vision économique.

4.3. Mise en place d'un Living Lab forestier

Afin de mettre en place un lieu d'échange permettant de faire cohabiter en harmonie les finalités économiques, écologiques et sociales dans le cadre de la forêt de la Grande Région, une approche de Living Lab a été initiée par le RND (RND, 1976). L'objectif est d'explorer des scénarios innovants de couplage de la mise en place d'un outil de surveillance de la forêt et d'un outil de suivi de sa fonction récréative. Les acteurs impliqués sont potentiellement tous les bénéficiaires de la forêt et relèvent tant de la sphère économique d'exploitation du bois et du tourisme que de citoyens se rendant régulièrement ou occasionnellement en forêt, comme l'illustre la figure 7.a.

Une idée directrice est d'éviter de limiter les affectations d'une zone forestière à un usage unique car cela exclut d'emblée de grandes zones et réduit les possibilités de synergies. Par exemple, l'affectation d'une grande surface à une pépinière dégrade à la fois le potentiel touristique et la biodiversité. Le but est plutôt de permettre à un même espace de cumuler plusieurs rôles mais de manière hiérarchisée selon ses caractéristiques et ses potentialités. Ainsi, la figure 7.b illustre une affectation forestière

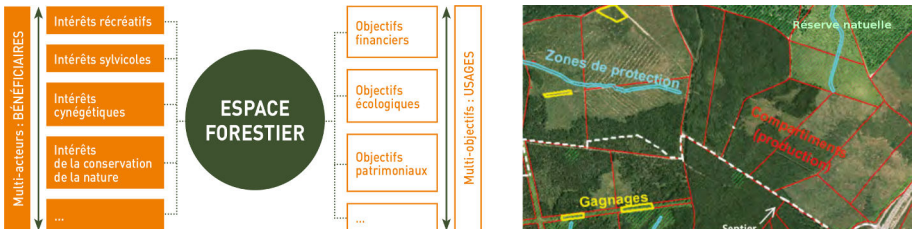


Figure 7. (a) Bénéficiaires et objectifs (b) Exemple d'affectation multiusage

combinant des compartiments de production, qui offrent des sentiers connectant des réserves naturelles et des zones de gagnages pour les animaux sauvages. S'il s'agit, à terme, d'aller vers un concept plus ouvert de forêt, à plus brève échéance, le but est surtout d'intégrer des données forestières (cadastre, imagerie satellite, fréquentation touristique, etc.) et d'étudier de nouveaux services ciblant en particulier la fonction récréative (fréquentation, mesure de services écosystémiques, etc.). Un défi de ce nouveau Living Lab est d'identifier et de mobiliser une communauté d'acteurs couvrant le spectre des bénéficiaires, en les faisant collaborer dans diverses phases itératives décrites dans la section 2. Certains acteurs interviennent plus en amont, au niveau de l'identification des scénarios innovants, et d'autres plus en aval sur la validation de prototypes intégrant diverses sources de données via les outils envisagés.

4.4. Exemples de scénarios et d'applications possibles

Différentes activités peuvent être évoquées et investiguées lors d'ateliers de co-création. Concernant le domaine récréatif, il peut s'agir des parcours découverte/santé, d'initiation à la cueillette de champignons, de l'étude de la faune/flore. Pour les propriétaires privés, il peut s'agir d'une aide pour la gestion de leur parcelle, par exemple, à l'aide d'une application dédiée. Un prototype d'outil peut être envisagé avec, dans un premier temps, le recours à l'existant. Par exemple, la figure 8. (a) montre une fonction d'isochrone proposée par le service ouvert OpenRoute (HeiGIT, 2018). Tandis que la figure 8. (b) illustre une application existante de gestion de parcelle forestière (Actiforet, 2018). Ces applications peuvent être utilisées telles quelles ou rapidement couplées via les mécanismes existants, par exemple, pour partager des informations sur la praticabilité de certains sentiers ou encore pour se mettre en relation avec des personnes disposant d'une parcelle ayant des caractéristiques proches en termes de surface et de peuplement.

Un cas concret récent concerne la gestion de la peste porcine qui sévit depuis 2018 dans cette région et qui touche les sangliers. Celle-ci a nécessité la mise en place d'un périmètre très contraignant d'isolement empêchant tout accès à la forêt. Le périmètre d'isolement nécessaire peut être affiné sur base de modèles et données cartographiques, des analyses d'images satellitaires par des géographes, des observations d'experts forestiers ou encore de citoyens circulant dans des zones autorisées. On peut ainsi imaginer des scénarios de gestion plus dynamique du périmètre de la

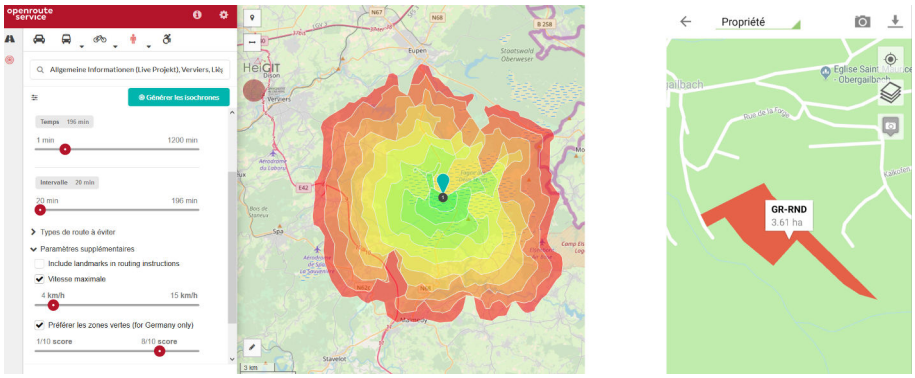


Figure 8. (a) Fonction isochrone pour balade en forêt (b) Gestion d'une propriété

zone afin d'assurer une certaine accessibilité touristique et exploitation économique tout en gérant le risque de propagation de l'épidémie.

5. Conclusions et perspectives

L'éco-conception logicielle dans le contexte de systèmes durables exige d'examiner à la fois comment le logiciel contribue aux objectifs de développement durable du système et comment il est lui-même durable. Divers outils ont pu être identifiés dans la littérature : l'analyse par objectifs, dirigée sur le développement durable, les approches participatives et d'éco-conception logicielle. Afin de les valider, nous avons utilisé deux cas d'étude. Un premier cas réalisé a posteriori a montré l'utilité de l'analyse par objectifs, de techniques d'optimisation, de conception architecturale, ainsi que la surveillance d'indicateurs prenant en compte la durabilité. Le second cas d'étude forestière, réalisé plus en amont, a permis de donner un éclairage centré plus spécifiquement sur la mise en œuvre d'un Living Lab.

Bien évidemment, ce travail ne prétend pas être exhaustif. Nous espérons cependant que ces deux illustrations issues du monde réel et analysées au moyen de plusieurs questions de recherche donnent un retour d'expérience utile à la mise en oeuvre des divers outils identifiés. Nous pensons aussi que la démarche d'analyse proposée et basée sur plusieurs questions de recherche peut s'appliquer plus largement comme guide à la fois pour les chercheurs et les praticiens ayant à réaliser le difficile exercice de proposer un soutien logiciel à un système afin d'en assurer la durabilité.

Nos travaux futurs continueront à s'appuyer sur les deux cas d'applications : pour le cas SAM, une évolution du système est envisagée pour intégrer des scénarios plus flexibles de mobilité. Les développements s'appuieront sur une approche de co-création et co-innovation. Concernant le second cas, nous suivrons l'évolution de la mise en place du Living Lab forestier du RND et assurerons l'animation des premiers accompagnements sur les scénarios esquissés. Les retours collectés continueront à consolider nos connaissances et à être partagés avec la communauté active en la matière.

Remerciements

Ce travail a été financé en partie par les projets SAMOBI et IDEES de la Région wallonne. Nous remercions SAM-Drive, l'asbl Ressources Naturelles Développement, les partenaires des projets RegioWoodII et Agreta, ainsi que les reviewers pour leur retour détaillé.

Bibliographie

Actiforet. (2018). *Ma forêt, carte et gestion.* https://infos.maforet.fr.

Ademe. (2012). *Économie circulaire : bénéfices socioéconomiques de l'écoconception et de l'écologie industrielle.* Ademe et vous, Stratégie & études ; no 33 2012-10-10.

Alam M. T., Porras J. (2018, 11). Architecting and designing sustainable smart city services in a living lab environment. *Technologies*, vol. 6, p. 99.

Banister D. (2008). The sustainable mobility paradigm. *Transport policy*, vol. 15, n° 2.

Becker C., Betz S., Chitchyan R., Duboc L., Easterbrook S. M., Penzenstadler B. *et al. (2016, Jan). Requirements: The key to sustainability.* IEEE Software*, vol. 33, n° 1, p. 56-65.*

Becker C. et al. (2015). Sustainability design and software: The karlskrona manifesto. In *Proc. of the 37th int. conf. on software engineering.* Piscataway, NJ, USA, IEEE Press.

Bordage F. (2015). *Eco-conception web - les 115 bonnes pratiques*. Eyrolles.

Cabot J. *et al. (2009, May). Integrating sustainability in decision-making processes: A modelling strategy. In* 31st int. conf. on software engineering - companion volume*, p. 207-210.*

Calero C., Piattini M. (2015). Green in software engineering. *Springer International Publishing.*

Djemame K. et al. (2014). Energy Efficiency Embedded Service Lifecycle: Towards an Energy Efficient Cloud Computing Architecture. In *Proc. of the 2nd Int. Conf. on ICT for Sustainability, Stockholm, Sweden.*

Geisberger R. *et al. (2008). Contraction hierarchies: Faster and simpler hierarchical routing in road networks. In* International workshop on experimental and efficient algorithms*, p. 319–333.*

Greenpeace. (2017). Clicking clean: who is winning the race to build a green internet? *http://www.greenpeace.org/usa/global-warming/click-clean.*

HeiGIT. (2018). Open route service. *https://openrouteservice.org.*

Hykes S. (2013). Docker - build, ship, and run any app, anywhere. *https://www.docker.com/.*

International Telecommunication Union. (2012). Recommendation Z.151 (10/12), User Requirements Notation (URN) – Language Def.

IUCN. (2006). The Future of Sustainability Re-thinking Environment and Development in the Twenty-first Century. *Report of the IUCN Renowned Thinkers Meeting.*

Johnson J. (2002). Keynote lecture. *XP Conference, Sardinia.*

Kusiak A. (2007, 01). Innovation: The living laboratory perspective. Computer-Aided Design & Applications*, vol. 4, p. 863-876.*

Lamsweerde A. van. (2009). Requirements engineering - from system goals to UML models to software specifications. *Wiley.*

Landtsheer R. D., Guyot Y., Ospina G., Germeau F., Ponsard C. (2018). Reasoning on sequences in constraint-based local search frameworks. In Integration of Constraint Programming, Artificial Intelligence, and Operations Research - CPAIOR, Delft, June 26-29.

Mahaux M., Heymans P., Saval G. (2011). Discovering sustainability requirements: An experience report. In REFSQ*, vol. 6606. Springer.*

Michel Renders. (2012). Sam-drive - shared shuttles for all. *https://www.sam-drive.be.*

Nagler R. et al. (2015). Sustainability and reproducibility via containerized computing. *Synchrotron Radiation*, vol. 4769, p. 145.

Nations U. (1987). *World commission on environment and development: Our common future.* Oxford Univ. Press http://www.un-documents.net/our-common-future.pdf.

OscaR. (2012). *OscaR: Scala in OR.* https://bitbucket.org/oscarlib/oscar.

Oyedeji S., Penzenstadler B. (2018). Karlskrona Manifesto: Software Requirement Engineering Good Practices. In *Proc. of the International Workshop RE4SuSy, Banff, Canada.*

Penzenstadler B. (2013). *What does sustainability mean in and for software engineering?* 1st Int. Conf. on ICT for Sustainability (ICT4S), Zürich: ETH E-Collection.

Penzenstadler B. *et al. (2014). Safety, Security, Now Sustainability: The Non-functional Requirement for the 21st Century.* IEEE Software*, vol. 31, n° 3, p. 40-47.*

Promotheus. (2018). From metrics to insight. *https://prometheus.io.*

RND. (1976). Ressources Naturelles Développement asbl. *http://www.rnd.be.*

Schliwa G. et al. (2015). Living labs and sustainability transitions – assessing the impact of urban experimentation.

Stefan D., Letier E., Barrett M., Stella-Sawicki M. (2011). Goal-oriented system modelling for managing environmental sustainability. In *3rd Int. Workshop on Software Research and Climate Change. Lancaster, UK.*

Tate K. (2005). *Sustainable Software Development: An Agile Perspective*. Addison-Wesley.

U4IOT. (2018). *Living Lab Methodology Handbook.* https://u4iot.eu/u4iot-tools.html.

Van Hentenryck P., Michel L. (2009). *Constraint-based local search*. MIT Press.

Vickery G. (2012). *Smarter and Greener? Information Technology and the Environment: Positive or negative impacts?* International Institute for Sustainable Development.

Williams A. *et al. (2017). Systems thinking: A review of sustainability management research.* Journal of Cleaner Production*, vol. 148.*

Younglai R. (2015). Rise of sharing services Uber, Airbnb points to a precarious labour climate. *The Globe and Mail http://bit.do/precarious-sharing-economy.*

Yu E. S. K., Mylopoulos J. (1997, avril). Enterprise modelling for business redesign: The i framework.* SIGGROUP Bull.*, vol. 18, n° 1, p. 59–63.*

Fouille de processus auto-définis : cas d'étude d'un moteur de recherche d'une bibliothèque numérique

Marwa Trabelsi [1], **Cyrille Suire** [2], **Jacques Morcos** [1], **Ronan Champagnat** [1]

1. *Université de La Rochelle - Laboratoire L3i La Rochelle*
   {nom.prenom}@univ-lr.fr
2. *Université Paris-Saclay, UVSQ, Laboratoire CHCSC*
   {nom.prenom}@uvsq.fr

RÉSUMÉ. *Un système de recherche d'information est un ensemble de ressources et d'outils permettant à des utilisateurs d'extraire des connaissances sur un ensemble de données. Dans de tels systèmes, les utilisateurs peuvent mener leurs recherches de diverses manières en fonction de leurs objectifs. Dans ce contexte, les processus suivis par les utilisateurs sont dits auto-définis. La signification, la structure et les résultats varient en fonction des pratiques. L'objectif de ce travail consiste à représenter le comportement des utilisateurs dans la recherche d'informations. Nous avons ainsi appliqué des algorithmes de fouille de processus sur les traces d'utilisation d'un moteur de recherche dans une bibliothèque numérique. Nous avons appliqué sept algorithmes sur deux ensembles de données correspondant aux interactions réelles des utilisateurs avec une bibliothèque numérique mettant à disposition des documents. Nous en déduisons ensuite des recommandations pour concevoir une méthodologie de fouille de processus auto-définis.*

ABSTRACT. *An Information Retrieval System is a set of resources and tools allowing users to search for information in a given domain. In such systems, users can have diverse ways to perform their research according to their objectives. In this context, users apply self-defined processes that may vary in term of significance, structure and results. Therefore, it would be useful to represent the behavior of users in their information seeking processes. To do so, we made use of process mining algorithms to mine user self-defined business processes in the context of user-Digital Library interaction. We considered seven process mining algorithms that we applied on two real datasets made of user traces corresponding to users' interactions with the Digital Library of Cultural Heritage. The main result is a set of recommendations to design a Process Mining methodology to mine user self-defined business processes.*

MOTS-CLÉS : *Bibliothèque numérique, Découverte des modèles de processus*

KEYWORDS: *Digital Library, Process Discovery*

---

1. Introduction

Les systèmes de recherche d'information se sont développés dans des entreprises de tous types et de toutes tailles. Les utilisateurs peuvent accéder aux informations et aux services via des applications web ou mobiles. Les processus fournis par les entreprises au travers de ces applications sont morcelés et laissent le plus souvent les utilisateurs déterminer seuls leur parcours pour atteindre leur objectif. Dans un tel contexte, le parcours de l'utilisateur en vue d'exécuter une tâche (achat d'un produit sur un site marchand, rechercher un document dans une bibliothèque numérique, *etc*.) est un « *processus auto-défini* ».

Parmi les systèmes qui produisent ce type de processus, le cas des bibliothèques numériques nous parait intéressant. En effet, les utilisateurs en situation de recherche d'information adoptent des comportements qui dépendent de nombreux facteurs, depuis leur niveau de connaissance du domaine jusqu'à leurs compétences spécifiques. Ces variables se traduisent par des processus de résolution d'une tâche de recherche d'information qui peuvent être variés. Afin d'étudier les différents parcours menant les utilisateurs à la résolution de leur tâche de recherche d'information et analyser les ressorts des différentes pratiques de recherche, nous utilisons les techniques de fouille de processus.

La fouille de processus (*Process Mining*) vise à extraire des connaissances liées aux processus à partir des journaux d'événements (logs). Elle offre divers moyens pour **découvrir (Process Discovery)**, **vérifier la conformité (Conformance Checking)** et **améliorer (Enhancement)** les processus réels (W. Van der Aalst, 2016). Chaque événement se réfère à une instance de processus (qui laisse une trace) et une activité. Les événements sont ordonnés et des propriétés additionnelles comme l'horodatage et l'utilisateur (p. ex. la personne ou la machine qui réalise l'activité) peuvent être ajoutées.

L'avantage principal des techniques que nous employons réside dans leur capacité à traiter le processus de recherche d'information dans son ensemble (un processus complet ayant une activité de début et une activité de fin). Elles diffèrent ainsi des techniques employées par exemple par Nouvellet *et al*. (2017) qui ont l'avantage de permettre une fouille de données des sessions utilisateurs, mais se limitent à la déduction d'un modèle fondé sur une chaîne de Markov d'ordre 1. Néanmoins, la question de la capacité de ces techniques de fouille de processus à analyser des processus auto-définis et à en déterminer la structure et les variations Luengo, Sepúlveda (2012) se pose.

Dans l'optique d'analyser les performances des techniques de découverte de processus appliquées aux processus *auto-définis* et de montrer leurs capacités à observer le comportement des utilisateurs suite à leurs interactions avec une bibliothèque numérique, nous présentons notre étude de cas dans la section 2, puis nous citons quelques

travaux connexes liés à l'application et la découverte des modèles de processus dans la section 3. La section 4 est dédiée à la description des différents algorithmes de découverte de processus. Ensuite, la section 5 décrit l'application des algorithmes sur nos jeux de données via l'outil ProM (Van Dongen *et al*., 2005). Elle présente également les métriques qui nous serviront pour l'évaluation des modèles obtenus. Nous fournissons également dans cette section une analyse qualitative des résultats. La section 6 conclue le travail et présente nos recommandations pour concevoir une méthodologie de fouille de processus auto-définis .

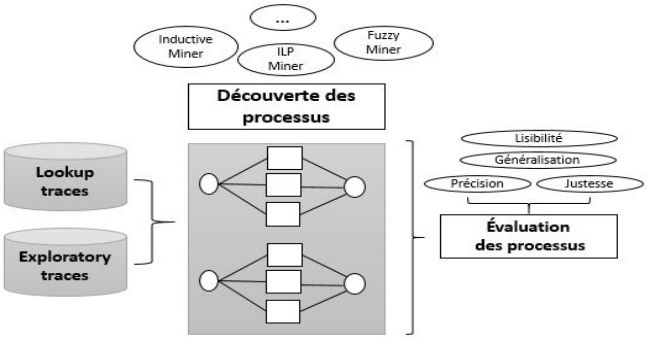


FIGURE *1. Découverte et évaluation des modèles de processus*

2. Présentation du cas d'étude

Les bibliothèques numériques sont des systèmes permettant de naviguer à travers de grandes masses de documents, souvent très hétérogènes. L'activité de recherche d'information qu'y mènent les utilisateurs est un processus (Cole *et al*., 2015) qui repose principalement sur les requêtes soumises au moteur de recherche, les filtres appliqués aux résultats et les documents consultés. Chacun de ces éléments dépend par ailleurs du type de tâche à l'origine de la recherche d'information, du niveau de connaissance de l'utilisateur pour le domaine de sa recherche et la manière dont sont indexées les données. Pour ces différentes raisons, l'usage d'une bibliothèque numérique par les utilisateurs produit beaucoup de processus avec des variations faibles.

Dans un tel contexte, l'étude des processus peut présenter un intérêt en vue de comprendre les principaux ressorts du comportement des utilisateurs en situation de recherche d'information. Nous proposons une comparaison des résultats de ces algorithmes en terme de performances vis-à-vis des données d'une part et de leur capacité à permettre à un regard expert de comprendre et analyser les processus détectés d'autre part (figure 1). Les données sur lesquelles nous nous appuyons sont issues d'expérimentations menées avec 40 étudiants débutant leur spécialisation dans le champ du patrimoine et de sa mise en valeur (Suire *et al*., 2016). Leur niveau de connaissance du domaine est homogène et suffisant pour qu'ils soient capables d'adapter leur comportement de recherche d'information en fonction de la tâche à accomplir. Le corpus de documents proposés par la bibliothèque numérique expérimentale contenait des documents hétérogènes, textes généraux, articles scientifiques, documents iconographiques et sources primaires, relatifs au champ de spécialisation des participants.

Lors de cette expérimentation, les participants ont été confrontés à des tâches de recherche d'information dans les deux grandes catégories définies par Marchionini (2006) et conçues par un spécialiste de la question du patrimoine. La catégorie *lookup* est constituée de tâches de recherche d'information simples relevant des sous catégories *recherche de faits* ou *recherche d'un document connu* des sous-catégories de Marchionini. Il s'agît par exemple de trouver une information précise telle qu'une date ou un lieu ou encore à trouver un document sur la base d'information connue. La catégorie *exploratory* implique quant à elle la sous catégorie *apprentissage et investigation* de Marchionini. Elle consiste en une tâche intellectuellement plus complexe visant à trouver l'information nécessaire pour répondre à une question ouverte telle que la construction d'une problématique de recherche et la préparation d'une présentation. Ces activités se traduisent dans les logs par des requêtes, ainsi que des informations relatives aux différents choix de navigation des participants dans les résultats et les pages de la bibliothèque numérique (contenu des requêtes, usage des filtres, sélection d'un document, accès au contenu en plein texte, etc.).

3. Travaux associés

De nombreux travaux ont été réalisés afin de découvrir des modèles de processus pertinents à partir des logs. Diamantini *et al*. (2016), par exemple, ont proposé le *Behavioral Process Mining* qui est une approche qui sert à identifier les sous-processus significatifs à partir des processus non structurés. Cette approche est fondée sur l'utilisation d'algorithmes de clustering hiérarchique appliqués aux graphes fouillés. Leur étude s'est appuyée sur des données réelles venant d'un CHU néerlandais. Ils ont identifié les traitements et les diagnostiques effectués les plus fréquents. De même, Song *et al*. (2008) présentent une approche générique qui utilise les techniques de classification. Leurs logs ont été divisés en sous-ensembles homogènes. Pour chaque sous-ensemble un modèle de processus a été créé. Ils ont introduit la notion de profil de logs afin de les comparer et de les caractériser.

À la suite de cette étude, d'autres travaux se sont concentrés sur l'application et l'évaluation des algorithmes de découverte de processus. Dans le domaine de la santé toujours, Mans *et al*. (2008) se sont focalisés sur ces aspects. En appliquant plusieurs techniques de découverte de processus à l'identification des parcours de groupes de patients, les auteurs ont démontré les avantages de l'utilisation des techniques de fouille de processus dans un cas réel. Ces travaux ont été étayés par d'autres, à l'image de Augusto *et al*. (2018), qui, sur la base d'une étude comparative des algorithmes de découverte des modèles de processus ont évalué les méthodes de découverte à travers neuf métriques. Ces travaux ont permis de conclure que les meilleures algorithmes, en termes de précision et de complexité, dans le cas des processus non structurés, étaient l'*Inductive Miner* et le *Genetic Miner*. D'autres travaux similaires se sont concentrés sur la construction d'un *framework* qui intègre les différents algorithmes de découverte afin de les évaluer (Jouck *et al*., 2018). Le but de ce framework est de trouver les algorithmes les plus performants en fonctions des logs. Dans le même cadre, Pérez-Alfonso *et al*. (2013) ont conçu une approche basée sur les propriétés des logs. Leur approche d'évaluation a été formalisée sous la forme d'un problème de classification.

Liu *et al*. (2017) utilisent d'autres méthodes comme l'analyse de séquences afin d'identifier les processus des étudiants dans un cours en ligne (MOOC). L'objectif de leurs travaux est d'identifier les comportements les plus signifiants des étudiants, afin d'aider les enseignants à adapter leurs stratégies pédagogiques aux différentes populations d'étudiants. Notre travail est similaire à ceux de Mans *et al*. (2008), Augusto *et al*. (2018) et Jouck *et al*. (2018) ; Nous avons appliqué plusieurs algorithmes de découverte de processus afin de distinguer leur efficacité et leurs limites. Dans notre étude, nous nous sommes intéressés aux *processus auto-définis* dans le contexte d'un moteur de recherche d'une bibliothèque numérique.

4. Découverte des modèles de processus

Les techniques de découverte de processus ont émergé dans les années 90 (W. Van der Aalst, 2012). Elles servent à générer des modèles de processus à partir d'une liste de traces d'exécution. Les processus ciblés étaient propres, complets et structurés. Au fil des années, les recherches concernant la découverte des processus se sont focalisées sur des problématiques de plus en plus complexes comme la génération de modèles à partir des traces d'exécution bruités, incomplets et/ou non structurés.

L'algorithme **Alpha (α)** est parmi les premiers algorithmes développés pour la découverte des modèles de processus. Il génère un Workflow-Net (un réseaux de Petri avec des caractéristiques spécifiques) qui traduit les processus enregistrés dans les logs W. Van der Aalst *et al*. (2004). L'algorithme α consiste à identifier les relations observées entre les activités dans les logs. Il se fonde sur les relations suivantes : succession immédiate, causalité, parallélisme et choix. L'algorithme α est incapable de découvrir des boucles ou des constructions avec choix non libres, ainsi que des activités dupliquées et des étapes intermédiaires. Afin de remédier à ces problèmes, l'algorithme α a été étendu dans un premier temps en α^{+} puis en α^{++} (Medeiros *et al*., 2004) ainsi que d'autres versions (Van Dongen *et al*., 2009). Cependant, ces algorithmes présentent toujours quelques limites parmi lesquelles leur sensibilité au bruit dans les logs. Dans cette étude nous considérons l'algorithme α^{++} uniquement.

L'algorithme **Heuristic Miner (HM)** a été développé par Weijters, Ribeiro (2011). Il se base sur une approche heuristique afin de résoudre les problèmes rencontrés par l'algorithme α. L'idée générale de cet algorithme consiste à identifier les ensembles de relations à partir des logs et de construire le modèle des processus en se basant sur les relations identifiées. La différence principale entre l'algorithme α et l'*Heuristic Miner* vient du fait que ce dernier se base sur des mesures statistiques afin de déterminer les relations entre les activités. L'algorithme est divisé en trois étapes. Il commence par produire un graphe de dépendance en comptant toutes les successions directes présentes dans les logs (les fréquences des successions sont ensuite stockées dans une matrice). Puis, il calcule le taux de dépendance entre chaque activité afin de ne conserver que celles qui ont une relation de causalité significative. Le graphe de causalité découvert (W. Van der Aalst *et al*., 2011) sera construit après avoir fixé un seuil sur le taux de dépendance et la fréquence des successions. La seconde étape consiste à identifier les divergences et les synchronisations en utilisant une approche

heuristique. Enfin, le graphe de causalité peut être transformé en un réseau de Petri si nécessaire.

L'algorithme **Inductive Miner (IM)** (Leemans *et al*., 2013) applique une approche basée sur le principe « diviser pour mieux régner ». Il divise les logs de départ en des sous ensembles de logs et, récursivement, il découvre le modèle de processus de chaque sous ensemble pour ensuite les combiner. De plus, il a la particularité de traiter le problème des traces non fréquentes (dont la fréquence d'apparition est rare vis-à-vis de l'ensemble totale des traces). Le modèle de processus obtenu est hiérarchiquement structuré sous la forme d'un *Process Tree* (Leemans *et al*., 2013). L'algorithme *Inductive Miner* est divisé en deux étapes : il commence d'abord par trouver le meilleur découpage à effectuer pour les logs ; ensuite, il détermine les opérateurs qui décrivent le mieux le découpage ; les deux étapes se répètent successivement sur les sous-ensembles. Le *Process Tree* miné peut être facilement transformé en une autre notation. L'*Inductive Miner* produit un modèle *sound* (W. Van der Aalst *et al*., 2004) qui ne contient pas d'anomalies et est capable de rejouer la majorité des logs, mais en contrepartie, il n'arrive pas à identifier des motifs complexes et non locaux de contrôle de processus.

L'algorithme **Genetic Miner (GM)** commence par créer aléatoirement un modèle de processus à partir des logs (W. M. Van der Aalst *et al*., 2005). La métrique de justesse de chaque processus est ensuite calculée. Par la suite, l'ensemble des modèles de processus construits évoluent à l'aide d'opérateurs génétiques : le croisement qui permet de combiner deux modèles et la mutation qui permet de modifier un modèle de processus. L'objectif de l'algorithme génétique est d'améliorer d'une façon itérative les modèles de processus jusqu'à atteindre un ensemble de modèle qui satisfait un critère d'arrêt. Cet algorithme retourne un modèle qui peut être converti en un réseau de Petri. L'algorithme *Genetic Miner* aborde les problèmes tel que le bruit, les traces incomplètes, les choix non libres, la concurrence et les activités dupliquées (W. Van der Aalst, 2016). Parmi les limites de cette approche, nous pouvons citer la complexité de construction des modèles de processus à partir des jeux de données réels.

Fuzzy Miner (FM) est un algorithme performant introduit par Günther (2009). Il est considéré comme une solution pour traiter le cas des processus métiers, issus de logs réels, en forme de spaghetti. La particularité de ces processus repose sur leur important niveau de flexibilité, qui réduit leur structuration. L'idée principale de cet algorithme consiste à construire une carte géographique simplifiée basée sur l'abstraction ou l'agrégation. Les activités et leurs relations (les arcs) peuvent être regroupés ou supprimés en fonction de leurs rôles dans le processus. L'algorithme *Fuzzy Miner* s'appuie sur deux concepts : la *signification* qui mesure l'importance relative de chaque activité, et la *corrélation* qui mesure la proximité de deux activités successives. L'importance de chaque activité peut être évaluée selon sa fréquence alors que la corrélation peut être définie en mesurant le temps d'occurrence entre deux activités. Les activités qui se produisent peu de temps l'une après l'autre sont fortement corrélées. À partir de ces deux concepts, il est possible de produire un modèle simplifié et cohérent en utilisant des heuristiques (Günther, 2009). Le *Fuzzy model* miné contient des *nœuds*

primitifs, qui ne contiennent qu'une activité et des *nœuds cluster* qui contiennent des activités agrégées. L'algorithme *Fuzzy Miner* offre aussi la possibilité aux utilisateurs de zoomer ou d'agréger le modèle.

Les algorithmes fondés sur les régions (**Region Based algorithms**) génèrent des modèles sous la forme de réseaux de Petri. Leur but est d'équilibrer la sur-précision (*overfitting*) et la sous-précision (*underfitting*) des modèles minés. Deux principaux algorithmes ont été proposés :

State Based Regions (SBR) (W. M. Van der Aalst *et al*., 2010) est un algorithme qui génère un réseau de Petri à partir d'un Système de Transitions (ST). Il offre la possibilité d'utiliser des abstractions afin de construire le ST. Les techniques de découverte de processus qui utilisent les abstractions sont capables d'obtenir leur modèle à partir des traces partielles contrairement aux techniques sans abstractions qui doivent utiliser l'ensemble complet des traces afin d'engendrer leurs modèles. De nombreux types d'abstractions ont été proposés pour l'algorithme SBR à l'instar des : *abstractions Set* (ni l'ordre ni la fréquence des activités dans les traces sont considérés), *Multi-Set* (seulement la fréquence des activités est prise en compte) et *Sequence* (l'ordre et la fréquence des activités sont considérés) et autre type d'abstraction dans lesquels chaque état du ST peut être représenté par une trace complète ou partielle. Le ST obtenu est ensuite transformé en un réseau de Petri en utilisant la théorie des régions. Cette dernière ne tient compte que des régions minimales non triviales comme les ensembles vides et les ensembles incluant tous les états du ST. Pour chaque région, l'algorithme SBR génère une place dans le réseau de Petri et pour chaque événement du ST, il génère une transition.

Tout comme le SBR, le but de l'algorithme **Language Based Regions (LBR)** est de trouver les places dans le réseau de Petri découvert. Il commence par traduire les logs en un *language process* (Werf *et al*., 2008). L'idée principale de l'algorithme est de trouver des places en se basant sur le *language process*. Toutes les places candidates correspondent à une région du langage. Le réseau de Petri découvert sera construit en ajoutant une place pour chaque solution non négative trouvée suite à la résolution d'un programme linéaire. Chaque solution est représentée sous la forme d'un triplet (X, Y, c), où X est l'ensemble des arcs d'entrée d'une place, Y est l'ensemble des arcs de sortie et c est le nombre de jetons dans la place. Enfin, l'algorithme LBR utilise les propriétés issues des logs (relations de causalité) pour déterminer le modèle final en assurant de ne mettre que les places expressives (ayant plus d'arcs entrants et sortants).

Les algorithmes basés sur les régions assurent la justesse (sous-section 5.2) dans la découverte des modèles de processus, ainsi que l'identification des structures de contrôles complexes. En revanche, ils sont incapables de traiter les logs incomplets et bruités.

5. Test des algorithmes de fouille de processus sur nos jeux de données

5.1. Jeu de données

Dans ce travail, nous utilisons deux jeux de données qui contiennent un ensemble d'activités pour les deux catégories de recherche *lookup* et *exploratory*. Le jeu de données *lookup* contient 102 traces, 1 655 événements et 34 utilisateurs alors que le jeu de données *exploratory* contient 29 traces, 1 472 événements et 29 utilisateurs.

Comme mentionné, les logs de la catégorie *lookup* contiennent 102 instances de processus (traces) de 80 variantes [1]. Six traces apparaissent plus d'une fois alors que 74 sont uniques. Dans les logs de la catégorie *exploratory*, il y a 29 traces qui sont toutes uniques. C'est une propriété importante des *processus auto-définis* où les traces récurrentes sont rares et peu redondantes contrairement aux processus structurés classiques, qui eux se répètent d'une manière identique. Les algorithmes de découverte de processus ont l'intérêt de trouver des modèles génériques à partir des traces non redondantes. Un autre aspect important des *processus auto-définis* par l'utilisateur vient de la répétition d'une activité au sein d'une même trace. Par conséquent, des traces longues peuvent être rapidement générées. Les boucles proviennent ainsi des hésitations des utilisateurs. Par exemple, trois traces dans les logs *lookup* contiennent plus de cinquante activités.

Tableau 1. Exemple de traces d'exécution

Case Id	User	Date	Label activity	Task category
1	USER1	2016-01-12|10:34:25	request	lookup
2	USER2	2016-01-12|10:36:25	request	lookup
1	USER1	2016-01-12|10:34:26	scroll	lookup
1	USER1	2016-01-12|10:34:28	ressourceAccess	lookup
3	USER3	2016-01-12|10:36:26	request	exploratory

5.2. Mesures d'évaluation : Conformité des modèles de processus

Pour évaluer la performance des algorithmes de découverte de processus, nous utilisons l'outil ProM (Van Dongen *et al*., 2005) pour qualifier les modèles obtenus. Tous les modèles ont été transformés sous la forme d'un réseau de Petri pour que l'on puissent vérifier la conformité des modèles par rapport aux logs de départ. D'après Rozinat *et al*. (2007); W. Van der Aalst (2016), un bon modèle de processus découvert doit trouver un équilibre entre la Justesse, la Précision, la Généralisation et la Simplicité. La plupart du temps ces critères s'opposent, améliorer l'un affaibli l'autre. Par conséquent, les algorithmes de découverte de processus doivent trouver un équilibre entre ces propriétés (Buijs *et al*., 2012; Adriansyah, 2014). Dans ce travail, trois métriques ont été employées :

– La **Justesse (J)** décrit le fait qu'un modèle de processus découvert correspond bien aux logs. Une façon de calculer la Justesse est de déterminer dans quelle mesure

1. une variante de processus correspond à la nature de la trace

les logs s'alignent avec le modèle obtenu (Adriansyah, 2014). Afin de présenter cette notion d'alignement, nous considérons une trace $L =< a, d, b, e >$ et deux modèles de processus M_1 et M_2. Nous considérons un tableau avec deux lignes où la ligne du dessus correspond à la trace de départ L et celle du dessous à la trace générée par le modèle. L'alignement optimal, γ_1, correspond au cas où il existe une trace L qui correspond exactement à une exécution de M_1.

$$\gamma_1 = \begin{array}{|c|c|c|c|} \hline \text{a} & \text{d} & \text{b} & \text{e} \\ \hline \text{a} & \text{d} & \text{b} & \text{e} \\ \hline \end{array}$$

Dans d'autres cas, si nous considérons que les traces générées par le modèle M_2 ne correspondent pas à la trace L de départ, il y aura des alignements multiples. Ainsi, avec cette méthode, nous pouvons distinguer deux sortes de mouvements. Des mouvements sur le modèle (*Moves on the Model*), qui consistent à modifier n éléments de la trace L pour correspondre au modèle M_2. Des mouvements sur la trace (*Moves on the Trace*), consistant à modifier n éléments de la trace générée par le modèle M_2 pour correspondre à la trace L (Adriansyah, 2014).

$$\gamma_{2a} = \begin{array}{|c|c|c|c|c|} \hline \text{a} & \gg & \text{d} & \text{b} & \text{e} \\ \hline \text{a} & \text{b} & \text{d} & \gg & \text{e} \\ \hline \end{array}$$

L'objectif des méthodes d'alignement est de trouver l'alignement optimal avec un coût réduit, c'est pour cette raison qu'un coût positif a été associé à chaque transformation (par exemple le symbole $\gg$). Après avoir appliqué l'algorithme d'alignement, la Justesse sera calculée sur chaque alignement afin de déterminer l'alignement optimal avec le meilleur taux de Justesse.

$$Justesse(L, M_2) = 1 - \frac{\delta(\lambda_{opt}^{M_2}(L))}{\delta(\lambda_{worst}^{M_2}(L))} \tag{1}$$

Où δ est la fonction de coût, $\lambda_{worst}^{M_2}(L)$ est le cas le plus défavorable où il n'y a aucune synchronisation possible entre la trace et le modèle. $\lambda_{opt}^{M_2}(L)$ représente les coûts obtenus pour chaque alignement optimal.

– La **Précision (P)** est une mesure qui indique si le modèle autorise uniquement le comportement observé dans les évènements de traces d'exécution. Un modèle découvert peut avoir des problèmes de sur-précision si sa valeur de précision est importante. Dans cet article nous avons utilisé la mesure de précision basée sur le résultat obtenu par l'algorithme d'alignement. Nous considérons que les logs et le modèle sont alignés (Adriansyah, 2014). La précision entre les logs T et le modèle découvert M est donné par :

$$Precision(T, M) = \frac{1}{|E|} \sum_{e \in E} \frac{en_T(e)}{en_M(e)} \tag{2}$$

Où E est l'ensemble des événements dans les logs T, A l'ensemble des activités, $en_T(e) \subseteq A$ est l'ensemble des activités présentes dans les traces et $en_M(e) \subseteq A$ l'ensemble des activités présentes dans le modèle.

– La **Généralisation (G)** est une métrique liée aux comportements non découverts. Le but est de découvrir un modèle de processus capable de généraliser des comportements qui ne sont pas apparus dans les logs. Cette métrique est utilisée pour identifier le problème de sur-précision. Dans le cas le plus favorable, le modèle découvert doit avoir une Généralisation faible et ne doit pas être restreint aux processus présents dans les logs (Adriansyah, 2014).

Les *Fuzzy model*s produits par l'algorithme *Fuzzy Miner* correspondent à une visualisation spécifique d'un processus. La qualité du modèle obtenu décrit le succès relatif de la procédure de simplification et la manière dont l'algorithme a pu extraire les informations les plus significatives. Afin d'évaluer la qualité des modèles découverts, deux mesures sont proposées : le Détail des nœuds et la Conformité (Günther, 2009) :

– Le **Détail des nœuds** décrit l'importance des activités visualisées dans le *Fuzzy model*, relativement aux activités agrégées ou supprimées. Les nœuds des activités visibles sont appelés nœuds explicites alors que les nœuds correspondant à un regroupement d'activité sont des nœuds implicites. La mesure de détail se calcule comme suit :

$$Detail = \frac{\sum v \in V^{s(v)}}{\sum n \in V^{s(n)}} \tag{3}$$

Où N correspond à l'ensemble des nœuds primitifs (activités présentes, agrégées ou supprimées) du *Fuzzy model* F, $V \subseteq N$ est le sous-ensemble de tous les nœuds visibles et s est une fonction qui associe à chaque nœud primitif F une mesure d'importance (Günther, 2009).

– La **Conformité** est une mesure qui décrit l'alignement entre le *Fuzzy model* F et les logs T. Le comportement enregistré dans chaque trace dans les logs est reproduit dans le *Fuzzy model*. Chaque activité dans les logs qui n'existe pas dans le *Fuzzy model* sera comptée comme une déviation. La conformité entre les logs T et le *Fuzzy model* F est définie par :

$$Conformite = \frac{M(T) - d + 1}{M(T) + 1} \tag{4}$$

Où M est le nombre total d'activités présentes dans les logs T et d est le nombre de déviations identifiées (Günther, 2009).

5.3. Discussion

Afin de calculer les mesures décrites dans la section 5.2, nous avons utilisé l'outil ProM. La mesure de Justesse a été calculée avec l'utilisation du package« *PNetReplayer* » alors que les mesures de Précision et de Généralisation ont été calculées à l'aide de package « *PNetAlignementAnalysis* ». Le tableau 2 présente les évaluations des algorithmes HM, IM, GM, LBR, SBR et α^{++}.

Tableau 2. Mesures de performance sur les modèles découverts

	Lookup			Exploratory		
	J	P	G	J	P	G
HM	0.00	0.00	0.00	0.00	0.00	0.00
IM	**0.9886**	0.2391	0.9994	**0.9315**	0.1437	0.9992
GM	0.9992	0.1800	0.9938	0.6232	0.8053	0.9963
LBR	0.6163	0.3825	0.9793	0.7835	0.1919	0.9622
SBR	0.8995	0.4233	0.9957	0.9560	0.2942	0.9918
α^{++}	0.00	0.00	0.00	0.00	0.00	0.00
FM	–	–	–	–	–	–

Comme on peut le lire dans le tableau 2, la Généralisation atteint toujours une valeur élevée quel que soit l'algorithme de découverte utilisé. Par contre, la Généralisation n'a pas pu être calculée pour les méthodes HM et α^{++}. De plus la Précision est toujours basse sauf pour l'algorithme GM dans le cas du jeu de données *exploratory*. La justesse, d'un autre côté, atteint son meilleur score avec l'algorithme IM sur les deux jeux de données.

Le réseau de Petri découvert avec l'algorithme α^{++} n'est pas un modèle *sound* car il contient des parties non vivantes. De plus, peu de traces ont été générées et aucune d'elles ne correspond à celles présentes dans les logs. Les mêmes limites ont été constatées pour l'évaluation des modèles générés par l'algorithme HM (cf. figure 2). Ces derniers ne sont pas aussi des modèles de processus *sound* puisqu'ils ne possèdent ni de marquage initial ni de marquage final. Le modèle ne présente pas de points de départs ni d'arrivée, En revanche, il est lisible et intéressant pour l'utilisateur final. Il est particulièrement utile pour observer les relations existantes entre les différentes étapes du processus. Si l'on tient compte du nombre d'itérations de chacune des étapes il permet d'établir rapidement le processus général et de comprendre l'origine des relations plus rares entre certaines étapes du processus.

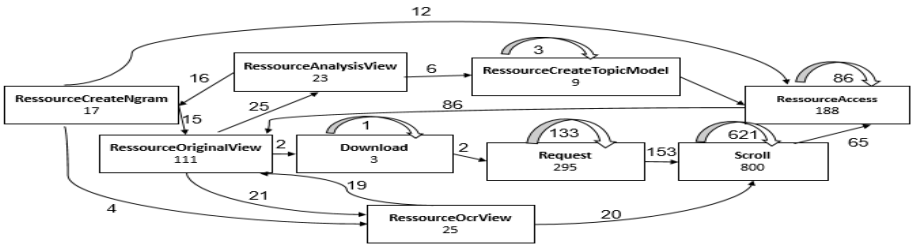


FIGURE *2. Algorithme Heuristic Miner appliqué à la catégorie exploratory*

La figure 3 présente le modèle de processus découvert par l'algorithme IM sur le jeu de données *lookup*. La mesure de Justesse est excellente sur les deux jeux de données *lookup* et *exploratory* (plus de $0,90$). De plus les modèles obtenus ont une haute Généralisation et une Précision basse.

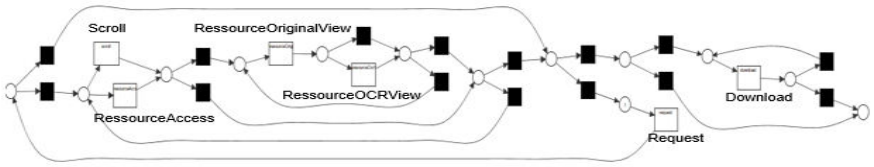


FIGURE *3. Algorithme Inductive Miner appliqué à la catégorie lookup*

Étant donné que ce modèle (fig. 3) est capable de détecter les points de départ et de fin du processus et qu'il élimine les arcs les moins importants, il donne une idée très claire du processus globalement suivi par les utilisateurs de la bibliothèque numérique. Les étapes intermédiaires et les arcs qu'il présente permettent de comprendre les différents cheminements des utilisateurs pour accomplir leur tâche. Il n'est cependant pas possible de mesurer la fréquence de chacune des options de navigation choisies par les utilisateurs et donc leur importance dans le processus.

Quand à l'algorithme GM, ce dernier découvre deux modèles avec une haute Généralisation. Le modèle du processus *lookup* donné par la figure 4 est capable de couvrir plus de logs que le modèle de processus *exploratory*. Malgré un long temps de calcul, la Justesse pour le jeu de données *lookup* atteint $0,9$ alors qu'elle baisse à $0,6$ pour le jeu de données *exploratory*. La mesure de Précision a donné un résultat opposé, le modèle découvert sur le jeu de données *lookup* est moins précis (avec $0,1$) et grimpe jusqu'à $0,8$ pour le jeu de données *exploratory* .

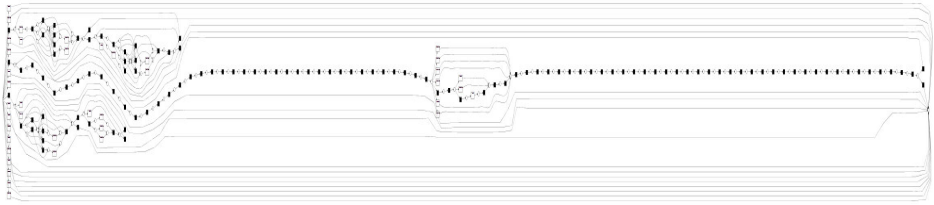


FIGURE *4. Algorithme Genetic Miner appliqué à la catégorie lookup*

La représentation produite par ce modèle (fig. 4) est extrêmement précise et couvre l'intégralité des chemins utilisés par les utilisateurs pour accomplir leur tâche. La contrepartie de cette précision est la grande complexité du graphe produit. Celui-ci ne permet pas une analyse qualitative et ne permet pas de réduire les processus individuels de chaque utilisateur à un ou plusieurs processus généraux. Il ne permet pas de déterminer les relations les plus importantes entre les différentes activités, ni les stratégies les plus globalement adoptées par les utilisateurs. Ce problème de lisibilité est une lacune importante des résultats que nous avons présentés pour ce modèle.

Concernant l'algorithme LBR, chaque transition des modèles minés est accessible lors de l'évaluation de la conformité, alors qu'ils ne possèdent pas de marquage initial

et ils adoptent une forme en fleur (c'est-à-dire que toutes les activités sont accessibles à partir d'un état de repos) (W. Van der Aalst, 2016). Pour les deux jeux de données, l'algorithme LBR produit des modèles avec une haute valeur de Généralisation et le taux de Justesse était dans les deux cas $\simeq 0,7$.

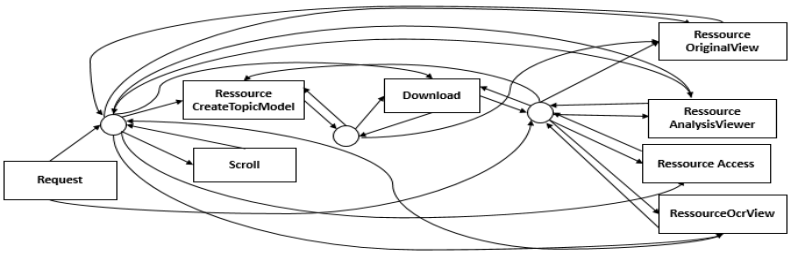


FIGURE *5. Algorithme LBR appliqué à la catégorie lookup*

En accord avec ces résultats, la représentation graphique produite dans la figure 5 propose une généralisation intéressante pour la lecture de l'expert. Les différentes activités sont bien représentées et les relations entre elles sont très détaillées. Cette représentation est surtout utile pour identifier les étapes de navigation intermédiaires et les options de navigation qu'elles proposent. Toutefois, à l'instar de la représentation produite par le modèle »Inductive Miner, la fréquence des différents choix effectués par les utilisateurs et leur importance dans la résolution de la tâche ne peuvent pas être observés.

Les modèles obtenus avec l'algorithme SBR ont été découverts en se fondant sur les options *Multiset abstraction* et le choix des traces partielles (section 4). Les deux réseaux de Petri découverts possèdent un marquage initial. L'algorithme SBR produit également deux modèles avec un niveau élevé de Généralisation. De plus, rejouer les logs sur les traces générées par les modèles obtenues produit des bons résultats que ce soit pour le jeu de données *lookup* que pour le jeu de données *exploratory* (Justesse $\simeq 0.9$). À l'instar de la représentation graphique issue de l'algorithme *Genetic Miner*, la représentation obtenue à l'aide de cet algorithme est très complète. Elle permet à l'expert du domaine de visualiser très finement tous les différents choix opérés par les utilisateurs pour naviguer d'une activité à une autre. Il présente cependant les mêmes défauts, telle que la difficulté à exploiter cette représentation dans un objectif de généralisation qui constitue une lacune importante dans les résultats des mesures comparatives que nous avons effectuées.

Tableau 3. Mesures obtenues pour l'algorithme Fuzzy Miner

	Détail des nœuds	Conformité
lookup	1.00	0.79
exploratory	1.00	0.84

Les mesures de qualité des *Fuzzy models* ont été fournies directement par le package *Fuzzy Miner*. Les résultats donnés par le tableau 3 montrent que les nœuds

primitifs visibles découverts dans les deux graphes sont à la fois importants et significatifs. De plus, rejouer les logs sur les traces issues des deux modèles minés donne de résultats satisfaisants (conformité $\simeq 0.8$). En corrélation avec ces résultats, ces modèles représentent une généralisation claire et intéressante pour les experts (fig. 6). La fréquence des choix faits par les utilisateurs est bien présentée et permet d'obtenir une vue globale des processus. Cependant, ces modèles ne permettent pas d'identifier les cas plus isolés et les processus les moins fréquents.

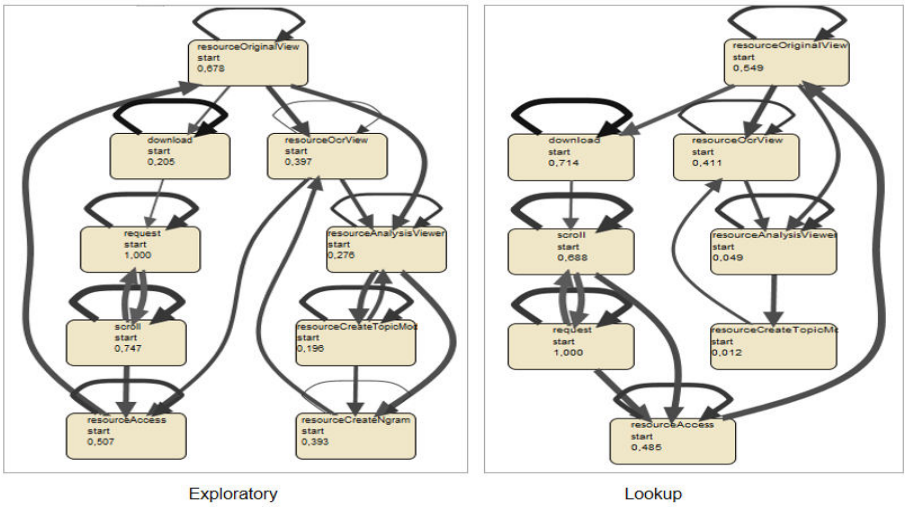


FIGURE *6. Algorithme Fuzzy Miner appliqué aux catégories lookup et exploratory*

6. Conclusion

Nous avons étudié la capacité des algorithmes de fouille de processus, à bâtir des modèles à partir des pratiques *auto-définies* des utilisateurs dans le contexte de la recherche d'information dans une bibliothèque numérique. Les algorithmes de fouille de processus α^{++}, *Heuristic Miner*, *Inductive Miner*, *Fuzzy Miner*, *Regions Based* et *Genetic Miner* ont été appliqués à deux jeux de données en utilisant l'outil ProM. Pour chaque algorithme, nous avons déterminé les mesures de Justesse, Précision et Généralisation afin de comparer les modèles minés et nous avons présenté une analyse qualitative des modèles découverts.

La principale conclusion de l'ensemble des mesures effectuées est que l'algorithme *Inductive Miner* donne de très bons résultats pour notre cas d'étude. Nous pouvons également constater que l'algorithme *Fuzzy Miner*, pour lequel il n'est pas possible d'appliquer les mêmes mesures, donne une très bonne vision globale des processus métier, avec une fonctionnalité de zoom et de ré-exécution des logs qui est d'un grand intérêt pour appréhender les phénomènes observés.

Par ailleurs, ce travail confirme la capacité des algorithmes fondés sur des méthodes de *clustering* à pallier les problèmes de variabilité des processus. Cette étude

nous fournit ainsi les fondements des travaux futurs ayant pour finalité de concevoir une méthodologie générique, capable de caractériser de manière pertinente les *processus auto-définis* par les utilisateurs d'applications numériques dans des contextes variés. Dans cet objectif, nous prévoyons de mener nos travaux dans trois directions : définir des métriques pour caractériser la variabilité des traces ; caractériser les variations induites par ces traces dans une famille de processus ; et définir une forme canonique d'écriture des *processus auto-définis* par l'utilisateur.

Bibliographie

Aalst W. Van der. (2012). Process mining: Overview and opportunities. *ACM Trans. on Management Information Systems*, vol. 3, n° 2, p. 7.

Aalst W. Van der. (2016). *Process mining: Data science in action*. Springer.

Aalst W. Van der, Adriansyah A., Van Dongen B. (2011). Causal nets: a modeling language tailored towards process discovery. In *Int. conf. on concurrency theory*, p. 28–42.

Aalst W. Van der, Weijters T., Maruster L. (2004). Workflow mining: Discovering process models from event logs. *IEEE Trans. on Knowledge & Data Eng.*, n° 9, p. 1128–1142.

Aalst W. M. Van der, De Medeiros A. A., Weijters A. (2005). Genetic process mining. In *Icatpn*, p. 48–69.

Aalst W. M. Van der, Rubin V., Verbeek H., Dongen B. F. van, Kindler E., Günther C. W. (2010). Process mining: a two-step approach to balance between underfitting and overfitting. *Software & Systems Modeling*, vol. 9, n° 1, p. 87.

Adriansyah A. (2014). *Aligning observed and modeled behavior*. Thèse de doctorat non publiée, Technische Universiteit Eindhoven.

Augusto A., Conforti R., Dumas M., La Rosa M., Maggi F. M., Marrella A. *et al*. (2018). Automated discovery of process models from event logs: Review and benchmark. *IEEE Transactions on Knowledge and Data Engineering*.

Buijs J. C., Van Dongen B. F., Der Aalst W. M. van. (2012). On the role of fitness, precision, generalization and simplicity in process discovery. In *Otm confederated international conferences" on the move to meaningful internet systems"*, p. 305–322.

Cole M. J., Hendahewa C., Belkin N. J., Shah C. (2015, mars). User Activity Patterns During Information Search. *ACM Trans. Inf. Syst.*, vol. 33, n° 1, p. 1:1–1:39.

Diamantini C., Genga L., Potena D. (2016). Behavioral process mining for unstructured processes. *Journal of Intelligent Information Systems*, vol. 47, n° 1, p. 5–32.

Günther C. W. (2009). *Process mining in flexible environments*. Thèse de doctorat non publiée, Technische Universiteit Eindhoven.

Jouck T., Bolt A., Depaire B., Leoni M. de, Aalst W. M. van der. (2018). An integrated framework for process discovery algorithm evaluation. *arXiv preprint arXiv:1806.07222*.

Leemans S. J., Fahland D., Aalst W. M. van der. (2013). Discovering block-structured process models from event logs containing infrequent behaviour. In *International conference on business process management*, p. 66–78.

Liu S., Fan H., Peng X., Liu Z., Cheng H., Sun J. (2017, 01). Mining learning behavioral patterns of students by sequence analysis in cloud classroom. *Int. Journ. of Distance Education Tech.*, vol. 15, p. 15-27.

Luengo D., Sepúlveda M. (2012). Applying clustering in process mining to find different versions of a business process that changes over time. In F. Daniel, K. Barkaoui, S. Dustdar (Eds.), *Bpm workshops*, p. 153–158. Berlin, Heidelberg, Springer.

Mans R. S., Schonenberg M., Song M., Aalst W. M. van der, Bakker P. J. (2008). Application of process mining in healthcare–a case study in a dutch hospital. In *International joint conference on biomedical engineering systems and technologies*, p. 425–438.

Marchionini G. (2006, avril). Exploratory Search: From Finding to Understanding. *Commun. ACM*, vol. 49, n° 4, p. 41–46.

Medeiros A. Alves de, Van Dongen B., Van Der Aalst W., Weijters A. (2004). Process mining: Extending the alpha-algorithm to mine short loops. *Univ. of Technology, Eindhoven*, vol. 113, p. 145–180.

Nouvellet A., Beaudoin V., D'alché-Buc F., Prieur C., Roueff F. (2017, novembre). *Analysis of gallica usage traces*. Research Report. Télécom ParisTech ; Bibliothèque nationale de France. Consulté sur https://hal.archives-ouvertes.fr/hal-01709264

Pérez-Alfonso D., Yzquierdo-Herrera R., Lazo-Cortés M. (2013). Recommendation of process discovery algorithms: a classification problem. *Res. Comput. Sci*, vol. 61, p. 33–42.

Rozinat A., Medeiros A. K. A. de, Günther C. W., Weijters A., Aalst W. M. van der. (2007). The need for a process mining evaluation framework in research and practice. In *International conference on business process management*, p. 84–89.

Song M., Günther C. W., Aalst W. M. Van der. (2008). Trace clustering in process mining. In *International conference on business process management*, p. 109–120.

Suire C., Jean-Caurant A., Courboulay V., Burie J.-C., Estraillier P. (2016). User Activity Characterization in a Cultural Heritage Digital Library System. In *Proc. of the 16th ACM/IEEE-CS on Joint Conf. on Digital Libraries*, p. 257–258. New York, USA, ACM.

Van Dongen B. F., De Medeiros A. A., Wen L. (2009). Process mining: Overview and outlook of petri net discovery algorithms. In *Transactions on petri nets and other models of concurrency ii*, p. 225–242. Springer.

Van Dongen B. F., Medeiros A. K. A. de, Verbeek H., Weijters A., Van Der Aalst W. M. (2005). The prom framework: A new era in process mining tool support. In *Icatpn*, p. 444–454.

Weijters A., Ribeiro J. (2011). Flexible heuristics miner (fhm). In *Computational intelligence and data mining*, p. 310–317.

Werf J. M. E. Van der, Dongen B. F. van, Hurkens C. A., Serebrenik A. (2008). Process discovery using integer linear programming. In *Icatpn*, p. 368–387.

Investigations of Process Mining Methods to discover Process Models on a Large Public Administration Software

Florent Mouysset[1,3], Célia Picard[2*], Christophe Bortolaso[1], Frederic Migeon[3], Marie-Pierre Gleizes[3], Christine Maurel[3] and Mustapha Derras[1]

1. Berger-Levrault, 64 Rue Jean Rostand, 31670 Labège, France
{first name}.{last name}@berger-levrault.com

2. ENAC, 7 Avenue Edouard Belin, 31400 Toulouse, France
{first name}.{last name}@enac.fr

3. IRIT, Université Toulouse 3 (Paul Sabatier),31400 Toulouse, France
{first name}.{last name}@irit.fr

ABSTRACT. Maintaining large and aging applications in a software house, with heterogeneous technologies, is very challenging. Whereas it is mandatory to continuously enhance user experience and maintain a good quality of service, the real business usage can be difficult to know precisely. To reach this goal, our project is to discover business models from the analysis of "logs". In this paper, we report on existing studies about applying process mining techniques and on our own experience with large datasets generated from daily end-user activities within an existing public services software. Our experiments led us to identify an interesting combination of features making our data hard to process with existing techniques. We conclude by providing perspectives to enable process discovery with such specific data.

KEYWORDS: Software logs analysis, Interweaving, Process Mining

1. Introduction

The complexity of software for public services increases with the constant evolution of regulations and with users' requirements. This has major impacts on software quality and maintenance processes. First, on the software editor side, this usually leads to an increasing number of bugs (Subramanyam & Krishnan, 2003). The scarceness of users' feedbacks collected by support teams, the heterogeneity of execution contexts and data make the bugs very hard to reproduce and to fix. As the project team designs the tests a priori, they may not reflect the users' practices. Thus, a gap appears between the theory and the reality (second Lehman's law (Lehman, 1980)), leading to failures on untested and unexpected cases. Then, from the user's point of view, the software becomes more difficult to use, interact with

* The work described in this paper was realized during Célia Picard's time at Berger-Levrault's.

and understand (Thompson *et al*., 2005). A new paradigm emerges and tries to design intelligent software, able of adapting themselves to specific users (Salinesi, 2017). This requires observing, collecting and understanding the users' activities. Logger systems constitute good solutions to collect users' activities. We made the assumption that logger systems combined with process mining techniques (Ailenei *et al.*, 2011) would enable the discovery of the observed system model. The discovered business process model would allow us, in a first step, to elicit use cases like user goals or business scenarios. In a second step, it would lead to prioritization of the maintenance, more realistic tests, and business intelligence.

In this paper, we report on experiments and lessons learned from applying process mining techniques on logs extracted from an existing on-use large public administration software. Our work can be compared to Astromskis's (Astromskis *et al*, 2015) who managed successfully to use process mining to understand user interaction on software. It led us to believe that their approach would be easily replicable on our dataset.

In the following pages, we first present our case study, the logged data and the associated software. Then, after presenting a general review of existing approaches, the best technique is selected to conduct experiments. Section 4 details these experiments. To conclude the paper, we stress that the presence of certain characteristics in our logs makes the process mining techniques ineffective. The problem is conceptual and not related to the logging system.

2. Case Study Description and Associated Data

We collected logs[1] generated by a rich client application used by town hall agents that we will call here "Software for public services" (SFPS). This software is composed of four modules. In this study, we focus on the analysis of logs coming from the Civil Status Management Module (hereafter called CSMM). This module contains numerous features related to civil registries, such as elections, births, deaths or court jurors. The software interface is composed of multiple tabs, each of them containing multiple forms. The navigation inside the CSMM is structured as follows: it starts with a home page containing a list of buttons. When clicking a button in this menu, a new tab opens and gets the focus. This new tab provides access to a second level menu containing itself user interface components to access forms and functional features.

The logging system is implemented in a software layer shared by all the different modules. This enabled developers to rely on a generic tracing system and to focus only on the development of functional features.

Despite the simplicity of the main navigation, the CSMM is extremely large and dense. Indeed, it is composed of about 600 different forms representing about

[1] Data are on open access on https://github.com/FM-BL/PublicCSMM_Logs

200,000 lines of code. We collected logs from 104 users over one year. The data contains 227,782 events for 60 mega-bytes in an XML structure.

```
<ExportList>
    <FormName>A</FormName>
    <FormDescription>A long description</FormDescription>
    <UserID>USER1</UserID>
    <OpeningTimestamp>2016-07-07T09:00:00</OpeningTimestamp>     A1
    <ClosingTimestamp>2016-07-07T10:00:00</ClosingTimestamp>     A2
</ExportList>
<ExportList>
    <FormName>B</FormName>
    <FormDescription>B long description</FormDescription>
    <UserID>USER1</UserID>
    <OpeningTimestamp>2016-07-07T09:15:00</OpeningTimestamp>     B1
    <ClosingTimestamp>2016-07-07T09:30:00</ClosingTimestamp>     B2
</ExportList>
<ExportList>
    <FormName>C</FormName>
    <FormDescription>C long description</FormDescription>
    <UserID>USER1</UserID>
    <OpeningTimestamp>2016-07-07T09:30:01</OpeningTimestamp>     C1
    <ClosingTimestamp>2016-07-07T09:59:59</ClosingTimestamp>     C2
</ExportList>
<ExportList>
    <FormName>D</FormName>
    <FormDescription>D long description</FormDescription>
    <UserID>USER1</UserID>
    <OpeningTimestamp>2016-07-07T09:35:00</OpeningTimestamp>     D1
    <ClosingTimestamp>2016-07-07T09:55:00</ClosingTimestamp>     D2
</ExportList>
<ExportList>
    <FormName>E</FormName>
    <FormDescription>E long description</FormDescription>
    <UserID>USER1</UserID>
    <OpeningTimestamp>2016-07-07T10:15:00</OpeningTimestamp>     E2
</ExportList>
```

Figure 1. Example of CSMM log

Each event represents the opening and the closing of a form. As depicted on Figure 1, the log is composed of a sequence of events each called "*ExportList*". Each event has five fields: "*FormName*" represents a unique form code and "*FormDescription*" contains a longer and more readable description; "*UserID*" indicates the user id performing the action; "*OpeningTimestamp*" and "*ClosingTimestamp*" are the form opening and closing dates.

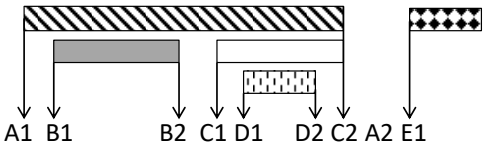


Figure 2. Temporal representation of the Figure 1 log

Figure 2 illustrates on a timeline the sequence of opening and closing events of the five forms present in Figure 1. Here, the form A enables to reach form D through form C. The closure of forms works with a FILO (i.e. First In Last Out) behavior.

However, the software interface is designed to allow other additional navigation instances. From a form D another form C can be opened in a new tab, both being

independent. Thus, it is possible to close the form D first, and after form C. Closing forms can also work with a FIFO (i.e. First In First Out) behavior. This indicates that a form closure does not necessarily imply a parent relationship. Furthermore, in some cases, the closing field ("*ClosingTimestamp*") might be missing (form E does not have a closing tag). It means either that the form encountered an error, making it impossible to write the closing date or that the form was not closed before leaving the application. Finally, we observed that sometimes, even if two forms are dependent, the closing date might be incoherent. For instance, the form C is child of form A (and are dependent, no new navigation instance) but the closing of form C occurs after the closing of its parent (A2). These three scenarios illustrate the wide range of behaviors that can be found in the processed data in our case study. The next section describes the different existing approaches that use input logs to discover process models.

3. Discovery Process: Existing Approaches

Multiple scientific domains focus on the analysis of human activity logs. In fact, several methods and algorithms designed to discover usage patterns and process models on business activity logs exist. We identified two types of relevant paradigms: 1) web mining and 2) process mining.

3.1. Web Mining

The web mining or web usage mining offers several approaches to analyze a web application and its usages (Srivastava *et al.*, 2000). Usually, the techniques are user-centric: all the events are related to a user. Existing techniques are most of the time based on sequence mining and pattern discovery methods. One of the typical purposes of web mining is to find the main way to accomplish a predefined goal. For instance, this purpose could be "What is the most frequent path to buy a specific product in an online shop?" With this kind of information, it becomes possible to optimize navigation paths and hence, increase sales (Spiliopoulou *et al.*, 2000). The goal of web mining differs from ours even though the data are similar. In web mining, the user is linked to a web session and each web session corresponds to a single use case (Srivastava *et al.*, 2000). Moreover, in the analysis the user goal is usually known beforehand: a purchase, a web page access or more generally the achievement of a well-known special action.

In our case, we do not have any information about users' goals in the current navigation path. Worse, the user can perform several use cases (i.e. pursue several goals) concurrently and we do not know how to relate the events to use cases. Thus, these techniques appear irrelevant for our problem.

3.2. Process Mining

Compared to the specificities of web analytics and mining, process mining appears to be a more suitable approach to our problem. The process mining is a recent scientific domain defined as a mix between data mining and workflow

analysis (van der Aalst, 2016). The process mining grasps three main aspects: 1) process discovery, 2) conformance model and 3) process enhancement. Since we seek to produce a process model from logs, we focused on the first aspect of the process mining: the process discovery. The process discovery is a collection of techniques that take logs as inputs and give process models as outputs. The input logs contain traces. Each trace is a sequence of events referring to the same case. Events are footprints left by the users when performing a business activity. Each event contains, at least, the name of the process activity, a timestamp and a specific process instance identifier[2]. Generally, the inputs are process-oriented (Pourmasoumi & Bagheri, 2017). This means that each trace corresponds to a process execution. Each process execution is identified by a unique id called the process instance id (or case id). The events belonging to the same process execution share the same process instance id: they are labelled. Logs containing labelled events are called labelled logs. The logs can contain various sorts of noises ((van der Aalst, 2016) p.148-151) and some data can be missing, incorrect or imprecise. This problem can occur in a continuous, intermittent or unpredictable manner. Business process models are generated as output. Various formalisms exist such as Petri Nets and Business Process Models and Notations (BPMN). Quality metrics can also be computed on these models. The most frequent metrics are the fitness (i.e. the ability to replay the behaviors seen in the logs) and the precision (i.e. the ability to forbid the behaviors unseen in the logs).

3.2.1. Approaches dealing with labelled logs

To enable process discovery, Cook and Wolf early proposed solutions based on neural-networks or Markov models, each solving a limited part of the problem. For example, the KTail approach finds a correct model but is noise-sensitive. On the contrary, the RNet methods find a less accurate model but are robust to noise (Cook & Wolf, 1998). However, none of those methods can manage concurrency aspects: in a business process, some activities can be performed simultaneously, and it is important to detect and represent these concurrent activities (Buijs *et al.*, 2012). The α-algorithm (van der Aalst *et al.*, 2004) brings a formalism to discover WorkFlow nets (WF-nets) (van der Aalst, 1998). WF-nets are a subclass of Petri Nets that can offer some interesting properties like soundness and safeness. They provide very good results, but these methods remain extremely noise-sensitive (van der Aalst *et al.*, 2004).

Probabilistic and statistical approaches are more robust and reliable than the α-algorithm (van der Aalst, 2016). The Heuristics Miner Algorithm (Weijters *et al.*, 2006), brings a significant improvement by solving problems such as the short loop discovery or the mining of long-distance dependencies (Wen *et al.*, 2006). Günther developed "The Fuzzy Miner" and provided techniques to discover processes from noisy data by using a graph model formalism (Günther & van der Aalst, 2007). With two new defined metrics (the significance and correlation) the method captures the

[2] In any case, an id is a unique identifier.

main patterns. But human intervention is required to tune parameter settings and identify the best model. Overall, probabilistic and statistical approaches are adapted to simple events but are limited when data is rich and complex.

The last family of approaches studied is the genetic algorithms. For instance, Alves de Medeiros's work (Alves de Medeiros, 2006) proposes to reuse the main concept of the genetic algorithms in process mining. With her method, an individual is considered as a process model on which selection and reproduction steps are applied. The main drawbacks of genetic algorithms are the long execution time and the local minimum risk. Moreover, genetic algorithms generate very complex models when using real-life logs (De Weerdt *et al.*, 2012).

3.2.2. Approaches dealing with unlabeled logs

Because obtaining labelled logs can be very challenging (Pérez-Castillo *et al.*, 2013), various methods try to deal with unlabeled data. The solution relies on labelling events artificially. We have identified two approaches: 1) making assumptions about structural, behavioral and temporal aspects of logged activities (Pourmirza *et al.*, 2015; Walicki & Ferreira, 2011), and 2) techniques based on experts' knowledge used to discover correlation rules and group events into traces (Pérez-Castillo *et al.*, 2013).

Pourmirza (Pourmirza *et al.*, 2015) proposes an algorithm to label logs coming from orchestration services. Three conditions must be respected: 1) it does not work with data coming from other services than acyclic orchestration services, 2) metrics on duration per activity must be given to the algorithm, and 3) the idle time between two requests should not be equal to the activity duration. Similarly, Walicki and Ferreira propose a probabilistic approach (Walicki & Ferreira, 2011) to process discovery with unlabeled logs. The approach has a high level of abstraction because the problem can be reduced to sequence mining challenges. Their technique tries to cover the sequence with a minimal set of patterns. However, some conducted experiments show that this approach is not suitable to process large logs (Walicki & Ferreira, 2011).

When the assumptions about data and processes are too variable, experts can explicitly write rules to inform the tagging algorithms. Pérez-Castillo (Pérez-Castillo *et al.*, 2013) propose a semi-automatized process able to build correlation rules. An event must contain various attributes and the logging system must be designed to put the candidate correlation attributes into events. An expert can choose the attributes in the log and determine the correlation rules.

3.3. Synthesis on process discovery

Broadly speaking, the literature shows that a large variety of techniques are available. Web mining is hard to apply because our users' goals are unknown. It is unclear in our software when and where use cases start and end. We chose to skip the early approaches of process mining for the more efficient recent ones. Process mining approaches dealing with unlabeled logs are not suitable as they require realistic temporal and structural assumptions. As described before, due to the

activity interweaving and interruptions, it is impossible to make any realistic statement about the users' activities in our logs. Our application is very large and has many legacy features; it is impossible to find experts with enough business knowledge and time to choose the attributes for each code injection. The difficulties of improving logging system are detailed in section 6.3. The logs only represent the opening and closure of forms, tabs and windows. The missing of the case id is the only limitation that forbids us to apply approaches dealing with labelled data. Thus, we artificially labelled our events to execute existing process mining techniques on them, as described on Astromskis's study (Astromskis *et al.*, 2015).

4. Experiments with Process Mining

Based on our data and literature review, process mining methods appeared to be promising approaches to process our logs and generate the software business process model. To generate the business process model of CSMM, we have conducted experiments with several process mining techniques implemented in ProM. ProM is an academic process mining software that offers several process mining techniques through plugins.

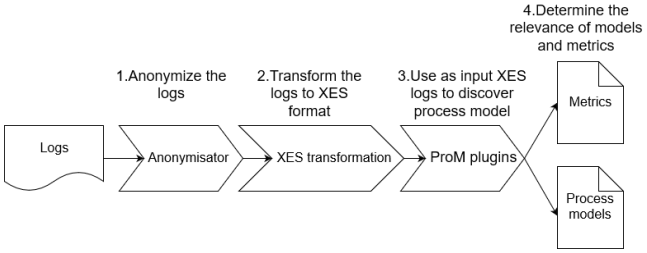


Figure 3. The main steps of our case study experimentation

We tested four different methods aiming to discover valid process models and compared the results. Hereafter, we describe our protocol and data transformation process, the tools, and finally our results.

4.1. Protocol

As described in Figure 3, our protocol is divided into three main steps: 1) anonymization, 2) XES transformation and 3) execution of ProM.

1. *Anonymization.* Due to privacy concerns, the first step consists in removing data that can be related to individuals such as: emails, addresses, names, surnames, etc. This information is replaced by IDs to maintain coherence and integrity in the logs. This process is reproducible and non-reversible.

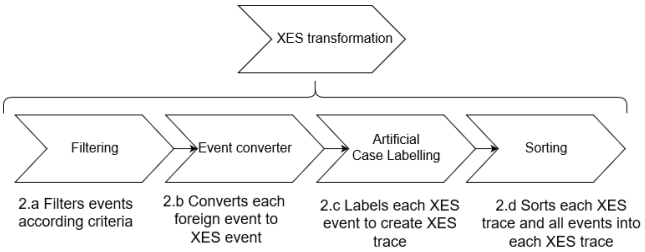


Figure 4. Sub-steps of the XES transformation

2. XES transformation. ProM accepts XES files as inputs. Therefore, a transformation from "CSMM" logs format to XES format is required. Labelling the events can be very difficult, thus we developed several strategies based on a successful study (Astromskis *et al.*, 2015). Each of them provides a smaller granularity supposed to give a better trace. We tested different transformations methods in the next sub-sections. Each concern is addressed as a sub-step as depicted in Figure 4.

2.a Filtering: we removed some noise and execution errors from the raw logs. More precisely, we identified and removed the events where the closing date is missing as they indicate a software failure, according to the development team. Overall, 7939 events were removed, corresponding to 3.3% of the total dataset.

2.b Event converter: various alternatives are available to translate CSMM events into XES events, each providing different advantages and semantics. We identified two strategies:

1) "1EGRC1XES" strategy consists in mapping each field of the CSMM event to the corresponding field of the XES event. This is the straightforward approach.

2) "1EGRC2XES" strategy considers that an XES event indicates either a form opening or a form closing. Thus, the conversion may produce two XES events for each event, one as a "start life-cycle transaction", the second as a "complete life-cycle transaction".

2.c Artificial Case Labelling: XES logs are structured into traces, which is a missing concept in our logs. In the XES format, a trace reflects a case which is not delimited in our data. Thus, we tried five different strategies to provide each event with a case id and artificially recreate traces from our logs:

1. The "Naive" strategy considers that all events have the same case id. Hence, only one trace is created containing all the events.

2. The "User Trace" (UT) strategy builds one trace per user. All events performed by the same user have the same case id.

3. The "User Trace by Day" (UTD) strategy considers the day scale and the user labelling; a case represents a user's daily activities. Hence, all the events performed by the same user in the same day have the same case id. Note that a case

cannot be performed over several days because town hall agents must respect working hours. Moreover, none of the use cases require several days to be accomplished.

4. The "User Home Trace by Day" (UHTD) considers the home page visit as the end of the previous business scenario and the beginning of a new one. All the events performed by the same user in the same day between two home page accesses have the same case id.

5. The "User Sub-Home Trace by Day" (USHTD) is based on the same principle as UHTD but considers the direct children forms of the home page. CSMM counts 11 sub-forms at a 2nd level of navigation. When an opening event of a sub-home form occurs, all the encountered events form a trace. All the events performed by the same user in the same day between sub-home forms have the same case id. We chose to stop at the second level because it still represents a manageable size of sub-nodes. The number of forms at the 3rd level increases exponentially.

We crossed all the possibilities between our two event conversion strategies and our five artificial use case labelling strategies. We decided not to consider the combination of USHTD and 1EGRC2XES strategies. In fact, we observed that by applying this strategy 50% of the events were located between accesses of forms. This led us to believe that this strategy would not provide any significant results. The artificial use case labelling strategies can be considered as cutting functions. In fact, all the opening forms of a considering level are cut points. The forms path before the cut belongs to a use case; the forms path after the cut belongs to another use case. We restricted our study to these five strategies because more refined approaches would over-cut the logs and would have created too many traces.

2.d Sorting: Finally, we sort the events by date in the resulting XES files. In the 1EGRC1XES condition, we sort the events by start date (i.e. opening of the form). In the 1EGRC2XES condition, we sort the events by the date kept in the event, thus opening date of the form for some events and closing date of the form for the others.

Execution. The last step consists in executing the four plugins, described in the apparatus below, on various data sets. Exactly, each one of the four listed plugins are tested with each generated XES file using the two different converters (2b) and the 5 different labelling strategies (2c), hence, a total of 4*2*5=40 experimentations.

4.2. Apparatus

To perform process mining, we rely on ProM v.6.7. ProM is an academic process mining software[3]. Various plugins exist, especially for process model discovery. As explained above we aim at testing and comparing four different methods on our data (see Table 1). The experiment has been carried on a standard laptop machine including 16Gb with 6Gb RAM dedicated for JVM, Intel® Core™ i5-4210M CPU@ 2.60GHz. Our evaluation is based on the replay fitness metric (Buijs *et al.*,

[3] http://www.promtools.org/doku.php

2012). The replay fitness indicates how much the discovered model can reproduce the behavior capture in logs. The replay fitness is ranged between 0 (the discovered model cannot replay any part of the logs) and 1 (a perfect replay).

Table 1. ProM Plugins Used and Tested

Name	Version	Output
HeuristicsMiner	6.7.70	Fitness metrics
Fuzzy	6.7.53	Replay percentage
Alpha Robust Miner	6.7.70	Petri Net. PN Conformance Analysis plugin to compute conformance
Evolutionary Tree Miner (ETM)	6.7.168	Process Tree / Display fitness per generation

4.3. Results

The results of the experiments are listed on Table 2. After the XES transformation step, we obtain traces corresponding to process runs in process mining. The trace number increased as the trace granularity became smaller. The event conversion produces a total of 227,782 events, considering only the opening of the forms (1EGRC1XES), and logically twice as much (455,564) when also considering closing events (1EGRC2XES). Overall, the results are disappointing, and no method allow us to discover a reliable process.

First, surprisingly, with the Heuristic Miner, the more we try to label use cases precisely, the lower the fitness is. We also observe that in over one-third of our tests, the Heuristic Miner could not provide positive fitness. Negative fitness indicates a very low success due to many remaining tokens in the associated Petri Net. The "1EGRC1XES" / "User Trace" condition provided the best result with a fitness of 0.55. For the Heuristic Miner, it appears simpler to deal with a unique high grain trace than to indicate various precise but noisy traces.

The ETM execution time was extremely long (several hours on our target computer), and the computation failed several times due to memory lack (i.e. java heap space). Even when reducing by 50% the dataset content, the best fitness provided by the ETM was only of about 0.59 (with Naïve labelling). This result is consistent with Weerdt's observation (De Weerdt *et al.*, 2012) that the genetics approaches are not very suitable to process real life data. On the contrary, the Alpha Robust Miner provides no result when applying the Naïve strategy. It becomes more efficient when splitting the data into use cases. It provides results between 0.40 and 0.46. We attribute this to a better noise robustness.

Finally, the Fuzzy miner provides the best results (~73%) with the "1EGRC1XES" / "UHTD" case. However, the discovered model is unreadable and generates a dense "spaghetti-like" process. In addition, a manual analysis of some traces replays shows that many transitions noted as wrong in the model exist. Unlike the Heuristic Miner, the Fuzzy miner is better with precise tagging techniques.

Table 2. Summary of experiment results

Converter	Labelling	Nb Traces	Nb Events	Heuristics Miner Fitness (0-1)	Fuzzy Replay %	ETM Fitness 0-1	Alpha Robust Miner replay score
1EGRC1XES	Naive	1	227 782	0.4084	61.77	0.59	-
1EGRC2XES	Naive	1	455 564	0.5351	47.07	0.59	-
1EGRC1XES	UT	104	227 782	0.5554	57.54	-	0.44
1EGRC2XES	UT	104	455 564	0.4036	43.80	-	-
1EGRC1XES	UTD	6 617	227 782	-1.6116	56.76	-	0.40
1EGRC2XES	UTD	6 638	455 564	-0.3839	39.68	-	0.45
1EGRC1XES	UHTD	12 794	227 782	-0.2543	73.03	0.51	0.41
1EGRC2XES	UHTD	12 840	455 564	0.2346	56.16	-	0.46
1EGRC1XES	USHTD	8 882	141 828	-0.1706	68.78	-	0.52

5. Discussion and threats of validity

None of the tested techniques provides satisfying results with our data, even though we tested multiple conditions and provided ProM with logs of various granularity of events and several use case labelling strategies. We found that some of the methods have trouble to simply provide results. This could be an implementation problem or a lack of memory, but our tests highlight that with large datasets, some algorithms require very powerful machines to reach their objective without causing issues. In addition, we observed that getting reasonably good results does not mean that the resulting graph is readable. Our example with the Fuzzy miner is quite a good illustration of this limit. Our software experts indicated a high level of wrong transitions on the generated models. This brings us to a well-known problem: the Oracle. How confident in our results can we be? Moreover, 73% of replay indicates that we still have 27% of noise.

Furthermore, despite the application of an equivalent methodology and protocol, our results are in opposition with the results obtained by Astromskis (Astromskis *et al.*, 2015). The main difference between their study and ours lies in the implementation of the logging system and the size of our software. We believe that the automatic and generic implementation of our logging system could be the main explanation to the differences between our conclusions and Astromskis et al.'s. In fact, the logging system was manually added in their work. This means that the logging software was designed a posteriori for their purposes. On the opposite, in our case, the logging system was designed and implemented long time before our research. Nevertheless, the lack of information on the complexity of the analyzed software is not sufficient to explain reliably our failure. Some of the characteristics of our data might also be a reason. We detail them in the next section.

6. Learned Lessons

After obtaining the results we conducted a qualitative analysis to understand why process mining techniques deprived us from satisfying results. Our analysis led us to understand many specificities about our data which were directly induced by the nature of the user interface and the users' practices. We illustrate the problem in the next sub-section with the secretary scenario.

6.1. The Town Hall Agent Scenario

The "town hall agent scenario" is a typical work case which explains the nature and complexity of our data. The scenario starts with a town hall agent who starts a business scenario (BS1) thus creating events. A phone call occurs interrupting the case; BS1 is suspended. During the phone call, the agent starts a new scenario (BS2) to fulfill the caller's request. This triggers the creation of some other log events. The phone call ends but the agent still must perform actions related to BS2. At this point, someone enters the office and makes a new request. Again, this interruption suspends the processing of BS2. The agent starts a third scenario (BS3). When it ends, the secretary can resume BS2 and produce associated events. Finally, BS2 ends and BS1 is resumed.

This sequence of actions is possible because: 1) software users can be interrupted while executing a use case and 2) the software allows to start one or several new use cases simultaneously. This flexibility between use cases and tasks is mandatory to provide efficient tools to support this kind of activities. The understandability of software logs consequently suffers from this type of practice and thus our ability to understand the tasks.

6.2. A combination of four features

Overall, following the previously described scenario, we have identified that our data is characterized by four features which, all combined, make the business model hard to discover:

Unlabeled Use Cases. The first issue is related to the impossibility of labelling use cases on the fly. The users require flexible software that allow multiple use cases in parallel thus there is no clear indication in the log about when a use case starts and stops. This is a strong tendency in modern-web apps, such as Single Page Applications, enabling more flexibility, interactivity and overall a better user-experience to end-users (Mesbah & van Deursen, 2007).

The Impossibility of making Temporal or Structural Assumptions. Our data provide no clear indication about the duration of tasks. In fact, depending on the situation, filling a form to update a civil status can take from a few minutes up to several hours due to the interruptions and the requests. Moreover, some of the forms and screens are used by multiple use cases. For example, declaring a newborn may require updating the civil status of a parent. This parent civil status update will be the same as for a new passport request. The nature of our logs does not enable to identify clearly which specific screen is used in which use case.

The Presence of Loops. The nature of the users' activities also involves many redundant activities. For instance, the agent can come back and forth between the home menu and the burial plot attribution form. This enables the users to fill forms in a row in order, for instance, to process a large quantity of records. This, again, is quite common in administrative activities and will have to be identified when applying analysis techniques on our logs.

Multiple Levels of Interweaving. As illustrated in the scenario on Figure 6, we found multi-level of interweaving between use cases. The logs represent a list of events where each one contains a starting timestamp and ending one. These timestamps might overlap with other ones in the entire content of the log. This interweaving is due to the ability of users to concurrently start, perform and close several use cases. Our logs do not contain one linear story but several interweaved situations. It appears that existing approaches have difficulties handling the secretary scenario depicted in our case.

These four features are today more or less manageable by different approaches. For example, Walicki (Walicki & Ferreira, 2011) is known to be suitable for interweaving and Pourmirza approach works for cyclic applications (Pourmirza *et al.*, 2015). However, the combination of these four features makes the problem very hard to solve.

6.3. Challenges with logging systems

We also investigated the logging system itself. Our analysis highlighted interesting aspects of the logging system and difficulties to enhance it. In our case study, the logging system is implemented in a transversal manner, ensuring a strong decoupling between the application modules. This enables developers to rely on a generic tracing system and to focus on functional features development. The logging system uses the minimal data provided, thus the event logs are inaccurate. To make the logs more detailed, the entire logging system would have to be redesigned; whereas it is financially impossible. This problem is not restricted to our application. In general, since the logging system is not a business feature, it is not a major concern for the production team. The fewer resources are involved to set up a logging system, the better. The existing logging systems can produce events with technical details used for debugging purposes but are not able to inform the case id necessary for process mining. Multi-layer applications increase the difficulty because the information needed to create the correct event is spread over several layers. Several logging systems may be mandatory, or a single central logging system, that may break with a decoupled architecture.

Finally, redesigning the logging system of a large part of legacy code may cause regression. In fact, in some specific situations, adding logging calls can have unexpected impact on performances or even on the reliability of the execution. This problem needs great attention, since most of the practical use cases are unknown, non-regression tests are potentially inefficient. This analysis of our logging system stresses the tension between being able to build a generic and transversal mechanism and collecting enough data to uncover the business models afterwards. This type of

issues solely appears when working on large, distributed applications, where modifying the code manually is not a viable option. Overall, this highlights the need to work on logging systems architecture to enable genericity and low granularity of collected events.

7. Perspectives and Conclusion

In section 2, we have indicated the main reasons making it extremely difficult to understand our software logs. These difficulties amount to matching the events of a process with the software task model. This matching can be formalized by a multi-criteria optimization problem where each event must be associated to a unique task instance according to the log history, and each task must try to be fully described by events. This leads to a huge search space due to the combinatorial explosion. For this reason, we believe that Artificial Intelligence approaches might be of interest to process software logs. These approaches can preprocess the log to obtain labelled logs and, hence, prepare the logs to the application of process mining techniques. Two domains are interesting: 1) the clustering and classification approaches like the artificial neural networks and support vector machines, 2) the Multi-Agent Systems (MAS), in particular the Adaptive Multi-Agent Systems (AMAS).

The first approach tries to manage and reduce the complexity and the non-structuration of some processes. Rocío (Rocío *et al.*, 2015) have done a systematic literature review on Process Mining with artificial neuronal networks and vector machines approaches. Only two studies propose a contribution on process discovery and only Song's work explicitly uses the model and workflow defined by Process Mining (Song *et al.*, 2013).

On the contrary, the second approach does not try to reduce or simplify the complexity of data but tries to grasp the entire complexity of a problem. Some MAS rely on emergent techniques to find solutions. For example, Adaptive Multi-Agent Systems (AMAS) (Di Marzo Serugendo *et al.*, 2011) have already proved efficiency to solve some complex problems like big data analysis (Belghache *et al.*, 2016) even in very noisy environments. AMAS exploits the collective intelligence of agents to organize themselves and build a business process model.

To produce a suitable product while more complexity is needed, future software will have to dynamically adapt themselves to the users. To design these systems, we cannot rely on a known business process model. As the system cannot be intrusive, activities user traces must be taken as input. Regarding our needs, the process mining seems to be able to provide models of our software from execution logs. Because our events are unlabeled and our logging system unmodifiable, we used similar preprocessing steps as existing studies. Then, a set of process mining techniques was executed including the ProM plugins. The results showed an important quantity of wrong transitions in the discovered processes. This result highlights the complexity of our data: interweaving of use cases, lack of use case labels and various loops creating wrong trace labelling. Because of the diversity of

users and use cases, we could not make temporal, structural, behavioral or semantic assumptions. To deal with the complexity and removing the noise, we consider exploring artificial intelligence approaches and particularly multi-agent systems as a potential solution.

Bibliography

Ailenei I., Rozinat A., Eckert A., van der Aalst W. M. P. (2011). Definition and Validation of Process Mining Use Cases. In *Proceedings of Business Process Management Workshops*, pp. 75-86, Springer Berlin Heidelberg, Clermont-Ferrand, France.

Alves de Medeiros A. K. (2006). *Genetic Process Mining.* Thesis in Computer Science, Eindhoven University of Technology.

Astromskis S., Janes A., Mairegger M. (2015). A process mining approach to measure how users interact with software: an industrial case study. In *Proceedings of the International Conference on Software and System Process*, pp. 137-141, ACM Press, New York, USA.

Belghache E., George J.-P., Gleizes M.-P. (2016). Towards an Adaptive Multi-agent System for Dynamic Big Data Analytics. In *Proceedings of the Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress*, pp. 753-758, IEEE, Toulouse, France.

Buijs J. C., van Dongen B. F., van der Aalst W. M. P. (2012). On the Role of Fitness, Precision, Generalization and Simplicity in Process Discovery. In *Proceedings of On the Move to Meaningful Internet Systems*, pp. 305-322, Springer Berlin Heidelberg, Rome, Italy.

Cook J. E., Wolf A. L. (1998). Discovering models of software processes from event-based data. *Transactions on Software Engineering and Methodology*, vol. 7, no 3, pp. 215-249, ACM.

De Weerdt J., De Backer,M., Vanthienen J., Baesens B. (2012). A multi-dimensional quality assessment of state-of-the-art process discovery algorithms using real-life event logs. *Information Systems*, vol. 37, no 7, pp. 654-676, Elsevier.

Di Marzo Serugendo G., Gleizes M.-P., Karageorgos A. (2011). *Self-organising software : from natural to artificial adaptation.* Springer.

Günther C. W., van der Aalst W. M. (2007). Fuzzy mining–adaptive process simplification based on multi-perspective metrics. In *Proceedings of International conference on business process management*, pp. 328-343, Springer, Brisbane, Australia.

Lehman M. M. (1980). Programs, life cycles, and laws of software evolution, *Proceedings of the IEEE*, vol. 68, no. 9, pp. 1060-1076, IEEE.

Mesbah A., & van Deursen A. (2007). An Architectural Style for Ajax. In *Proceedings of Conference on Software Architecture*, pp. 9-9, IEEE, Mumbai, India.

Pérez-Castillo R., Weber B., Piattini M. (2013). Correlation of Business Activities Executed in Legacy Information Systems. In *Proceedings of Evaluation of Novel Approaches to Software Engineering*, pp. 48-63, Springer, Warsaw, Poland.

Pourmasoumi A., Bagheri E. (2017). Business process mining. *Encyclopedia with Semantic Computing and Robotic Intelligence*, vol. 1, no 1, World Scientific Publishing Company

Pourmirza S., Dijkman R., Grefen P. (2015). Correlation Mining: Mining Process Orchestrations Without Case Identifiers. In *Proceedings of Service-Oriented Computing*, pp. 237-252, Springer, Goa, India.

Rocío A., Maita C., Martins L. C., Ramón C., Paz L., Peres S. M., Fantinato M., Chopra A., Singh B. J. (2015). Process mining through artificial neural networks and support vector machines: A systematic literature review. *Business Process Management Journal*, vol. 21, no 6, pp. 1391-1415, Emerald Group Publishing Limited.

Salinesi C. (2017). Un jour, les Systèmes d'Information se concevront eux-mêmes. In *Proceedings of INFormatique des ORganisations et Systèmes d'Information et de Décision*, pp. 5-6, INFORSID, Toulouse, France

Song M., Yang H., Siadat S. H., Pechenizkiy M. (2013). A Comparative Study of Dimensionality Reduction Techniques to Enhance Trace Clustering Performances. *Expert Systems With Applications*, vol. 40, no. 9, pp. 3722-3737, Elsevier.

Spiliopoulou M., Pohle C., Faulstich L. C. (2000). Improving the Effectiveness of a Web Site with Web Usage Mining, In *Proceedings of Web Usage Analysis and User Profiling*, pp. 142-162, Springer, San Diego, CA, USA.

Srivastava J., Cooley R., Deshpande M., Tan P.-N. (2000). Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data. *SIGKDD Explorations Newsletter*, vol. 1, no. 2, pp. 12-23, ACM.

Subramanyam R., Krishnan M. S. (2003). Empirical analysis of CK metrics for object-oriented design complexity: implications for software defects. *Transactions on Software Engineering*, vol. 29, no. 4, pp. 297-310, IEEE.

Thompson D. V., Hamilton R. W., Rust R. T. (2005). Feature Fatigue: When Product Capabilities Become Too Much of a Good Thing. *Journal of Marketing Research*, vol. 42, no. 4, pp. 431-442, American Marketing Association.

van der Aalst W. M. P. (2016). *Data Science in Action*. Springer.

van der Aalst W. M. P. (1998). The Application of Petri Nets to Workflow Management. *Journal of Circuits, Systems and Computers*, vol. 8, no. 1, pp. 21-66, World Scientific Publishing Company.

van der Aalst W. M. P., Weijters T., Maruster L. (2004). Workflow Mining: Discovering Process Models from Event Log. *Transactions on Knowledge and Data Engineering*, vol. 16, no. 9, pp. 1128-1142, IEEE.

Walicki M., Ferreira D. R. (2011). Sequence Partitioning for Process Mining with Unlabeled Event Logs. *Data & Knowledge Engineering*, vol. 70, no. 10, pp. 821-841, Elsevier.

Weijters A. J. M. M., van der Aalst W. M. P., Alves de Medeiros A. K. (2006). *Process Mining with the Heuristics Miner-Algorithm*. Technische Universiteit Eindhoven, Tech. Rep. WP, vol. 166, pp. 1-34.

Wen L., Wang J., Sun J. (2006). Detecting Implicit Dependencies Between Tasks from Event Logs. In *Proceedings of Frontiers of WWW Research and Development*, pp. 591-603, Springer, Harbin, China.

Modélisation de la variabilité des indicateurs dans le cadre des administrations de services publics

Diego DIAZ[1], Mario CORTES-CORNAX[1], Agnès FRONT[1], Cyril LABBE[1], David FAURE[2]

1. Univ. Grenoble Alpes, CNRS, Grenoble INP[1], LIG, 38000 Grenoble, France
prenom.nom@univ-grenoble-alpes.fr

2. Groupe INCOM & COSI+
dfaure@incom-sa.fr

RESUME. *Les administrations de services publics comme les distributeurs d'eau font appel à des éditeurs logiciels pour implémenter leurs processus métier et calculer des indicateurs exigés par décideurs et les autorités de régulation. Cependant, dans un même métier, un processus peut avoir plusieurs variantes et dans ce cadre, le calcul et la définition des indicateurs de performance deviennent complexes pouvant prendre la forme du recalcul d'un indicateur existant, de la création d'un nouvel indicateur, voire même de l'impossibilité de le calculer. Cet article propose des éléments permettant de faciliter la définition et le calcul d'indicateurs clés de performance, prenant en compte différentes variantes d'un processus.*

ABSTRACT. *Administrations of public services such as water distributors call upon software publishers to implement their business processes and key performance indicators (KPI) required by decision makers and regulation entities. However, in the same business, a process can have several variants and, in this context, the KPI definition and calculation are complexified, having to recalculate an existing KPI, creating a new one or even the impossibility of providing it. This paper proposes elements allowing to define and calculate KPI by considering different reference process' variants.*

Mots-clés : Indicateur Clé de Performance, Processus métier, Variabilité.

KEYWORDS: *Key Performance Indicators, Business Process, Variability.*

1. Introduction

Les administrations de services publics comme les distributeurs d'eau (DE), font appel à des éditeurs logiciels afin d'implémenter leurs *Business Process* (BP) et de calculer leurs indicateurs de performance. Chaque DE s'organise selon certaines règles générales adaptées aux savoir-faire propres. Tous les DE partagent les mêmes

[1] Institute of Engineering Univ. Grenoble Alpes

BP de référence, mais dans certaines situations concrètes, les processus diffèrent en raison des caractéristiques spécifiques des services (taille du département, type de service…). Chacune de ces variantes est un ajustement du BP de référence (Reichert *et al.,* 2015) et par conséquence, les processus de chaque DE doivent être évalués différemment.

Notre étude est un cas industriel sur la variabilité de calcul des indicateurs de performance dans les familles de processus métier des distributeurs d'eau faisant appel à l'éditeur logiciel INCOM[2]. Cet éditeur travaille pour plus de 130 distributeurs de services publics et dispose d'une suite logicielle de sept applications supportant leurs gestions métier. Chaque distributeur doit donc évaluer ses propres BP sous différents critères, lesquels n'ont pas la même définition d'un distributeur à l'autre. En conséquence, INCOM construit et examine séparément tous les indicateurs clés de performance stratégiques et opérationnels, appelés *Key Performance Indicators* (KPI), pour garantir la conformité du processus de référence aux règles métier de chaque distributeur et aux règles imposées par les autorités de régulation (entités d'audit). INCOM doit donc fournir des KPIs à chacun des distributeurs tout en respectant leurs exigences, leurs règles générales (ex. lois d'administration publiques) et leurs règles spécifiques (ex. fonctionnement interne).

Actuellement, la modélisation de la variabilité des BP est généralement supportée par les *Business Process Model Families* (BPMFs) (La Rosa *et al.,* 2011), lesquelles ne tiennent pas compte de la variabilité des KPIs en fonction des choix réalisés et des critères définis par les parties prenantes. Nous abordons cette problématique par la question suivante : comment modéliser et supporter la variabilité du calcul des KPIs dans une famille de BP ?

Dans cet article, la section 2 présente le cas d'étude. Les travaux liés à la variabilité des BP et à la modélisation des KPIs sont discutés en section 3. La section 4 conclut et présente les perspectives envisagées.

2. La variabilité des KPIs au sein d'INCOM

Nous illustrons la variabilité des KPIs au sein d'INCOM à travers le BP de référence *Créer Contrat.* INCOM configure et déploie indépendamment ce BP pour chacun des DE et doit ainsi leur fournir des indicateurs pertinents selon les exigences des parties prenantes (cf. figure 1). Chacune des variantes du BP *Créer Contrat* constitue un ajustement du processus métier de référence dû à l'organisation interne des DE et à certaines règles générales adaptées au savoir-faire propre.

La figure 1 montre la sensibilité des KPIs à la variabilité. Un même KPI peut avoir plusieurs variantes selon un contexte spécifique. Par exemple pour le KPI de référence *Nombre de E-Contrats Actifs* (i.e., nombre de contrats électroniques), une partie-prenante peut en réalité vouloir calculer le nombre de contrats signés par mail (NECA 2), une autre le nombre de contrats demandés et signés par le Portail (NECA 3), alors qu'une entité d'audit exigera le nombre de contrats demandés par Mail ou Portail et

[2] http://www.incom-sa.fr/

signés par le Portail (NECA 6). Ainsi, il est possible de distinguer trois types de variantes : i) la configuration souhaitée par le distributeur d'eau (NECA 2 et 3) ; ii) l'interprétation et la définition des parties prenantes (NECA 1 et 4) ; iii) les critères d'évaluation du service de l'entité d'audit (NECA 5 et 6).

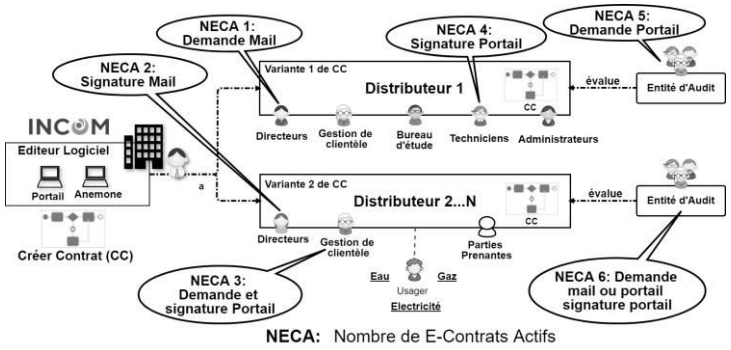


Figure 1 Variabilité d'un KPI au sein de la structure de gestion des DE

La variabilité du BP peut être spécifiée à l'aide d'un modèle de caractéristiques utilisé pour modéliser la variabilité des lignes de produits logiciels (Gröner et al., 2011), dont deux éléments sont souvent utilisés dans la construction de ce modèle : i) relation parent-enfant *obligatoire* ou *facultative* ; ii) regroupement en groupes *Or* et *And* (Batory, 2005). Si un DE souhaite configurer le processus *Créer Contrat*, il doit forcément activer la fonctionnalité *Concevoir Contrat.* En revanche, il a le choix d'activer ou non les fonctionnalités : *Activer Contrat Directement, Envoyer et Traiter Contrat* et *Activer Contrat Signé,* comme l'illustre la figure 2 a).

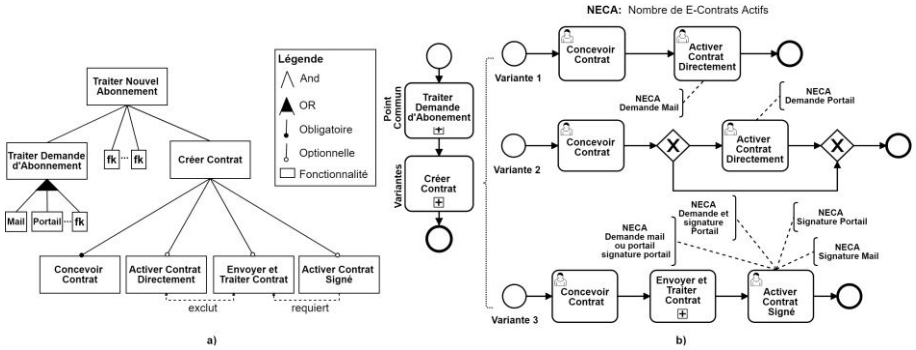


Figure 2 a). Choix des caractéristiques du processus Traiter Nouvel Abonnement ; b). Variantes pour le sous-processus Créer Contrat

Si trois DE décident de configurer le processus *Créer Contrat* différemment, nous aurons en conséquence trois variantes du processus de référence, comme l'illustre la figure 2 b). Le KPI peut ainsi être calculé par rapport aux moyens utilisés pour faire la demande d'abonnement, par rapport aux moyens utilisés pour signer le contrat ou

bien par rapport à une combinaison des deux concepts. Un KPI de référence a donc plusieurs variantes à calculer lors de la configuration, le déploiement et l'exécution du BP. Ces variantes peuvent impliquer la création de nouveaux KPIs et/ou le recalcul du KPI de référence. Par exemple, pour le KPI de référence *Nombre de E-Contrats Actifs* (**NECA**), ils existent les variantes : *Nombre de E-Contrats Actifs dont la Signature a été effectuée par le Portail* (**NECASP**) et *Nombre de E-Contrats Actifs dont la Demande a été effectuée par Mail* (**NECADM**)

3. Approches supportant la variabilité des processus métier et calcul de KPIs

La variabilité des BP est définie comme la capacité d'exprimer et de produire différentes variantes d'un BP à partir d'un modèle configurable (Reichert *et al.,* 2015). La modélisation de la variabilité des BP est généralement supportée par les *Business Process Model Families* (BPMFs) (La Rosa *et al.,* 2011), également connues comme les processus métier configurables (Rosemann et van der Aalst, 2007). Les BPMFs fournissent une représentation globale des BP d'un domaine donné grâce à une modélisation systématique de la variabilité et des points communs des BP récurrents connue sous le nom de *Business Process Model Template* (BPMT) (Gottschalk *et al.*, 2009). Un BPMT consiste donc en une collection de BP pour toute une famille de manière superposée (Czarnecki et Antkiewicz, 2005).

Afin de modéliser et d'exprimer les variantes des processus, plusieurs approches ont été proposées : PROVOP (Hallerbach *et al.,* 2010) et PESOA (Schnieders et Puhlmann, 2006) sont probablement les plus exploitées (Cognini *et al.,* 2016). Il existe également d'autres approches comme : C-EPC (Rosemann et van der Aalst, 2007), ConfBPMF (Ognjanovic *et al.,* 2012), C-iEPC (La Rosa *et al.,* 2011), SOSPLs (Mohabbati *et al.*, 2011), vBPMN (Döhring et Zimmermann, 2011), CPM-SPS (Lönn *et al.,* 2012), C-YAWL et CPM (Gottschalk *et al.*, 2009), BPFM (Cognini *et al.,* 2016), CMMN (OMG, 2016) et Declare (Pesic *et al.,* 2007).

Ces approches supportant les BPMFs permettent principalement d'inclure des variantes pour les BPMTs ayant été, a priori, complètement déterminés, i.e., incluant toutes les variantes et relations possibles de la famille de processus. Toutefois, ces approches sont complexes à appliquer si les entités à considérer ne sont pas connues a priori (Cognini *et al.,* 2016), ce qui est le cas des entités dépendant de caractéristiques spécifiques du DE, n'ayant pas de variante connue dans le BPMT. Au contraire, l'approche BPFM permet d'inclure facilement des variantes d'affinement même si le processus a déjà été configuré. Cela représente un avantage car les informations propres au contexte de déploiement du BP sont prises en compte.

De surcroît, aucune de ces approches ne considère l'impact que les BPMFs provoque sur le calcul des KPIs, ne permettant pas d'aligner les BP configurés avec les stratégies métier. Les approches comme CPM et CPM-SPS se basent sur l'approche C-YAWL dont la représentation ne permet pas d'inclure les objets de données d'entrée ni de sortie, contrairement à l'approche BPFM. Les objets de données sont utiles dans l'affinement des variantes après la configuration du BP en prenant en compte les informations du contexte d'exécution. De plus, l'approche BPFM est la seule à supporter la variabilité au niveau de la structure organisationnelle,

ne supposant pas un BPMT entièrement spécifié. En outre, les approches CMMN et Declare se basent sur le paradigme déclaratif, ne permettant pas d'inclure des variantes d'affinement après la configuration du BPMT, ni un ordre précis d'exécution du BP configuré, contrairement à l'approche BPFM.

Dans d'autres domaines, les architectures classiques de *Data Warehouse, Business Intelligence, Business Activity Monitoring* ou *Business Performance Management* (Golfarelli *et al.*, 2004) répondent entre autres à l'importance de faire respecter les objectifs définis par les stratégies métier via le calcul de métriques. Néanmoins, dans le cas de BP variables, il ne suffit pas d'extraire les informations à partir de données métier, d'autant plus lors de définitions différentes des KPIs et de critères d'évaluation flexibles. Il est donc nécessaire que ces architectures disposent d'un composant de variabilité des BP et des KPIs afin de surveiller, mesurer et évaluer tous les BP opérationnels et stratégiques configurés membres d'une même famille, pour ainsi permettre aux décideurs de mettre en place des opérations tactiques afin d'ajuster leurs actions en fonction de leurs stratégies.

Nous constatons que l'approche BPFM est la seule à permettre i) de modéliser la variabilité de BP avant et après la configuration du BPMT comme c'est le cas dans les distributeurs de services publics ; ii) d'inclure des objets de données contenant les informations pertinentes pour le calcul des variantes des KPIs. C'est pourquoi nous proposons d'étendre la notation du BPFM pour identifier, modéliser et supporter les variantes des KPIs dans une seule collection, en fonction des choix souhaités par le distributeur avant et après la configuration du BPMT, de l'interprétation des parties prenantes et es critères d'évaluation de l'entité d'audit.

4. Conclusion et travaux futurs

Afin de fournir des KPIs à chaque distributeur de services publics, l'éditeur logiciel INCOM doit tenir compte de leurs règles générales, de leurs règles spécifiques, de leurs définitions des métriques et de leurs critères d'évaluation. Nous proposons d'intégrer le calcul d'indicateurs clés de performance dans des processus métier configurables, i.e. dans un BPMT permettant de dériver les variantes d'un processus de référence à partir de la configuration des instances, pour ensuite modéliser la variabilité des KPIs.

Nos travaux futurs portent sur la conception d'une méthode de modélisation de la variabilité du calcul des KPIs dans des familles de processus métier. Cette méthode de modélisation permettra de définir les variantes d'un KPI de référence lors de la configuration d'un processus métier et ensuite lors de son déploiement et son exécution. L'objectif est d'aider à la conception de nouveaux KPI et/ou à leur recalcul vis-à-vis d'un KPI de référence selon : i) la configuration souhaité par le distributeur ; ii) l'interprétation et la définition des métriques par les parties prenantes ; iii) les critères d'évaluation du service de l'entité d'audit.

Bibliographie

Batory, D. (2005). Feature models, grammars, and propositional formulas. In *International Conference on Software Product Lines* (pp. 7–20). Springer.

Cognini, R., Corradini, F., Polini, A., & Re, B. (2016). Business process feature model: an approach to deal with variability of business processes. In *Domain-Specific Conceptual Modeling* (pp. 171–194). Springer.

Czarnecki, K., & Antkiewicz, M. (2005). Mapping features to models: A template approach based on superimposed variants. In *International conference on generative programming and component engineering* (pp. 422–437). Springer.

Döhring, M., & Zimmermann, B. (2011). vBPMN: event-aware workflow variants by weaving BPMN2 and business rules. In *Enterprise, Business-Process and Information Systems Modeling* (pp. 332–341). Springer.

Golfarelli, M., Rizzi, S., & Cella, I. (2004). Beyond data warehousing: what's next in business intelligence? In *Proceedings of the 7th ACM international workshop on Data warehousing and OLAP* (pp. 1–6). ACM.

Gottschalk, F., Wagemakers, T. A. C., Jansen-Vullers, M. H., van der Aalst, W. M. P., & La Rosa, M. (2009). Configurable process models: Experiences from a municipality case study. In *International Conference on Advanced Information Systems Engineering* (pp. 486–500). Springer.

Gröner, G., Wende, C., Bošković, M., Parreiras, F. S., Walter, T., Heidenreich, F., … Staab, S. (2011). Validation of families of business processes. In *International Conference on Advanced Information Systems Engineering* (pp. 551–565). Springer.

Hallerbach, A., Bauer, T., & Reichert, M. (2010). Capturing variability in business process models: the Provop approach. *Journal of Software Maintenance and Evolution: Research and Practice*, *22*(6-7), 519–546.

La Rosa, M., Dumas, M., Ter Hofstede, A. H. M., & Mendling, J. (2011). Configurable multi-perspective business process models. *Information Systems*, *36*(2), 313–340.

Lönn, C.-M., Uppström, E., Wohed, P., & Juell-Skielse, G. (2012). Configurable process models for the Swedish public sector. In *International Conference on Advanced Information Systems Engineering* (pp. 190–205). Springer.

Mohabbati, B., Hatala, M., Gašević, D., Asadi, M., & Bošković, M. (2011). Development and configuration of service-oriented systems families. In *Proceedings of the 2011 ACM Symposium on Applied Computing* (pp. 1606–1613). ACM.

Ognjanovic, I., Mohabbati, B., Gaevic, D., Bagheri, E., & Bokovic, M. (2012). A metaheuristic approach for the configuration of business process families. In *Services Computing (SCC), 2012 IEEE Ninth International Conference on* (pp. 25–32). IEEE.

OMG. (2016). Case Management Model and Notation (CMMN), *1.1*(December).

Pesic, M., Schonenberg, H., & Van der Aalst, W. M. P. (2007). Declare: Full support for loosely-structured processes. In *edoc* (p. 287). IEEE.

Reichert, M., Hallerbach, A., & Bauer, T. (2015). Lifecycle management of business process variants. In *Handbook on Business Process Management 1* (pp. 251–278). Springer.

Rosemann, M., & van der Aalst, W. M. P. (2007). A configurable reference modelling language. *Information Systems*, *32*(1), 1–23.

Schnieders, A., & Puhlmann, F. (2006). Variability Mechanisms in E-Business Process Families. In *Proceedings of the 9th International Conference on Business Information Systems (BIS'06)* (Vol. 85, pp. 583–601).

Une approche situationnelle pour la définition et l'adaptation d'une méthode d'évolution logicielle pilotée par les données

Jolita Ralyté

ISS, CUI, Université de Genève
Battelle bâtiment A, 7 Route de Drize, 1227 Carouge, Suisse
jolita.ralate@unige.ch

RÉSUMÉ. Vous trouverez ici une courte présentation de l'article intitulé "A Situational Approach for the Definition and Tailoring of a Data-Driven Software Evolution Method" co-écrit avec X. Franch et al., et publié dans les actes de la conférence CAiSE 2018.

Mots-clés : évolution logicielle, ingénierie de méthodes situationnelles, processus logiciel

1. Introduction

Le succès de l'évolution des logiciels dépend en grande partie de la sélection des fonctionnalités appropriées à inclure dans la prochaine version. Une telle sélection est difficile et les entreprises relatent souvent de mauvaises expériences concernant l'acceptation des utilisateurs. Pour surmonter ce défi, il existe un nombre croissant d'approches proposant une utilisation intensive des données pour conduire l'évolution. Cette tendance a motivé la création de la méthode SUPERSEDE[1], qui propose la collecte et l'analyse des commentaires des utilisateurs et des données de surveillance comme base de référence pour déterminer et hiérarchiser les exigences, qui sont ensuite utilisées pour planifier la prochaine version. Cependant, chaque entreprise peut être intéressée par l'adaptation de cette méthode en fonction de facteurs contextuels tels que la taille du projet, la portée, l'implication des utilisateurs, etc. Notre objectif a été de définir SUPERSEDE en tant que méthode de référence pouvant être adaptée à différentes situations. Par conséquent, nous avons adopté l'ingénierie de méthodes situationnelles (IMS) (Henderson-Sellers et al., 2014) comme une approche de base pour concevoir la méthode SUPERSEDE et guider son adaptation à un contexte particulier.

[1] SUPERSEDE – projet EU H2020 (www.supersede.eu)

2. SUPERSEDE en tant que méthode situationnelle

Le processus de la méthode SUPERSEDE s'inspire de la boucle de contrôle autonome proposée pour les systèmes adaptatifs (Brun, et al., 2009) et inclus quatre étapes : la collecte de données, leur l'analyse, la décision concernant les nouvelles exigences d'évolution du logiciel, et l'opérationnalisation des décisions. A l'origine, chaque étape de la méthode avait été formalisée en plusieurs éléments : activités, artefacts, rôles et outils. Les activités sont des tâches qui impliquent un ou plusieurs rôles et sont exécutées à l'aide d'outils, qui reçoivent et produisent un ou plusieurs artefacts. Des exemples d'activités sont la collecte de commentaires (la phase de collecte), la définition d'une ontologie de domaine (analyse), la hiérarchisation des exigences (décision) et la planification de la mise en production (opérationnalisation).

Dans ce travail nous avons proposé une redéfinition systématique de la méthode SUPERSEDE en termes d'un catalogue de composants de méthode en suivant les principes et techniques d'IMS. En particulier, en s'inspirant de l'approche proposée dans (Mirbel et Ralyté 2006) nous avons défini le metamodel pour les composants de méthode de SUPERSEDE. En fait, chaque activité avec ses rôles, artefacts et outils liés est devenu un composant de méthode. Le concept de cadre de réutilisation proposé par Mirbel et Ralyté (2006) a servi de base pour définir un ensemble de critères permettant de décrire le contexte dans lequel chaque composant peut être sélectionné et utilisé. Finalement, nous avons suivi l'approche d'IMS par assemblage de composants (Ralyté et al., 2003) pour définir le processus permettant de guider la définition d'une personnalisation de la méthode SUPERSEDE dans un contexte particulier. L'application de la nouvelle mouture de la méthode SUPERSEDE a été testée et approuvée dans un cas de développement et d'évolution d'une plateforme de ville intelligente développé par un partenaire industriel.

Dans cet article nous avons également fait une contribution à la théorie d'IMS en proposant une nouvelle manière de définir le contexte de réutilisation des composants de méthode, notamment en utilisant les modèles de buts étendus avec des annotations contextuelles – une extension de iStar2.0 (Dalpiaz et al. 2016).

Bibliographie

Brun Y. et al. (2009). Engineering Self-Adaptive Systems through Feedback Loops. *Software Engineering for Self-Adaptive Systems*, Springer.

Dalpiaz F. et al. (2016). iStar2.0 Language Guide. https://arxiv.org/abs/1605.07767

Franch X. et al. (2018). A Situational Approach for the Definition and Tailoring of a Data-Driven Software Evolution Method. *CAiSE 2018*, LNCS 10816, Springer, pp. 603-618.

Henderson-Sellers B. et al. (2014). *Situational Method Engineering.* Springer.

Mirbel I., Ralyté J. (2006). Situational Method Engineering: Combining Assembly-based and Roadmap-driven Approaches. *Requirements Engineering Journal*, 11(1): 58–78.

Ralyté J., Deneckère R., Rolland C. (2003). Towards a Generic Model for Situational Method Engineering. *CAiSE 2003*, LNCS 2681, Springer, pp. 95-110.

Recommandation d'itinéraires

Recommandation et valorisation d'objets patrimoniaux

L. Rajaonarivo [1], A. Fonteles [2], C. Sallaberry [1], P. Roose [1], MN. Bessagnet [1], P. Etcheverry [1], A. Lacayrelle [1], C. Marquesuzaà [1], C. Cayere [1], Q. Coudert [1]

1. Université de Pau et des Pays de l'Adour, LIUPPA, France
prenomcomplet.nom@univ-pau.fr

2. Indiana Wesleyan University, USA
prenomcomplet.nom@indwes.edu

RÉSUMÉ. *Nous présentons un travail qui, dans un contexte de promotion du tourisme dans les Pyrénées, vise à valoriser les atouts méconnus relatifs au patrimoine culturel matériel (architectural et mobilier) et au patrimoine culturel immatériel (traditions, événements sociaux). L'objectif est double. Il s'agit tout d'abord de constituer une base de données patrimoniale fédérant des données géoréférencées hétérogènes puis, de proposer des services de diffusion de ces données dans le cadre de différents scénarios de valorisation. Ces services permettent notamment d'alimenter des applications touristiques Web et mobiles. Nous allons présenter des propositions visant la recommandation de points d'intérêts (POI) patrimoniaux dans des scénarios d'usage touristique.*

ABSTRACT. *In a context of valorization of the tourism in the « Pyrénées » area in France, we present a research work that aims at the use of unknown assets related to the tangible cultural heritage (architectural and furniture) as well as the intangible cultural heritage (traditions, social events). The goal is twofold. First of all, we have to design and to build a cultural heritage oriented database federating heterogeneous georeferenced data. Then, we have to propose data dissemination services according to various valorization scenarios. These services allow to supply Web and mobile tourism applications. The papuer focuses on the recommandation of cultural heritage Points of Interest (POIs) within touristic scenarios.*

MOTS-CLÉS : *système de recommandation hybride, point d'intérêt (POI), contexte, génération d'itinéraires*

KEYWORDS: *hybrid recommender system, point of interest (POI), context, travel itinerary generation*

1. Introduction

Le projet de recherche Européen FEDER TCVPYR (2017-2020) a pour objectif un inventaire du patrimoine bâti et du patrimoine culturel immatériel de la villégiature et du thermalisme dans le massif pyrénéen français. Il vise le développement de la connaissance et de la fréquentation des Pyrénées à travers ses atouts patrimoniaux. Ce riche patrimoine qui reste à inventorier, est autant architectural que paysager et culturel. Dans le cadre de ce projet pluridisciplinaire regroupant entre autres historiens, géographes, anthropologues, pour la partie informatique, l'objectif est double : (1) constituer une base de données fédérant des points d'intérêts (POI) patrimoniaux référencés par des experts, et (2) proposer des services de valorisation de ces POI dans le cadre de différents scénarios de valorisation. Ces services permettent (i) d'exporter tout ou partie des POI vers l'Open Data afin de les rendre accessibles à tous et/ou (ii) d'alimenter des applications touristiques. Nous avons identifié deux principaux challenges. Le premier réside dans la gestion de l'hétérogénéité des différentes sources de données que nous utilisons. Par exemple, les chargés d'inventaire (dépendants des Conseils Régionaux) utilisent actuellement deux logiciels différents (RenablLP2 et Gertrude [1]), s'appuyant respectivement sur un modèle de données spécifique pour stocker les données du patrimoine. De manière similaire, pléthore de modèles sont utilisés dans le monde de l'Open Data (Foursquare [2], Datatourisme [3], Wikivoyage [4], WikiData [5]). Pour répondre à ce premier challenge, nous avons proposé un modèle unifié permettant d'intégrer toutes ces données (Bessagnet *et al*., 2018) et (Fonteles *et al*., 2018). Le second challenge sur lequel ce papier se focalise est lié à la valorisation des données du patrimoine stockées dans notre base de données unifiée. Nous proposons un ensemble d'algorithmes dédiés à la génération d'itinéraires « contextualisés » (i.e., des séquences de POI à visiter) adaptés aux utilisateurs d'une application mobile. Cette recommandation d'itinéraires prendra en compte le profil de l'utilisateur, sa position, les caractéristiques de son appareil ainsi que d'autres paramètres tels que le temps disponible pour effectuer la visite des POI ou le moyen de transport. Une recommandation pourra également être enrichie par des informations extraites de l'Open Data. Nous proposons une approche basée sur la notion de recommandation hybride. L'originalité de cette approche consiste à rapprocher des POI déjà parcourus par le touriste de ceux parcourus par d'autres touristes afin de mettre en exergue et d'évaluer des POI candidats selon le principe « un autre touriste qui a visité les mêmes POI que vous a aussi visité les suivants ». Ce principe est mis en œuvre dans notre approche grâce à l'intégration d'un algorithme d'optimisation de colonies de fourmis (*ACO: Ant Colony Optimization*) largement utilisés dans la recommandation

1. RenablLP2 et Gertrude sont des outils de création et de gestion de dossiers électroniques appliqués à la documentation de l'inventaire du patrimoine culturel. RenablLP2 est utilisé en région Occitanie et Gertrude l'est en région Nouvelle Aquitaine et Occitanie.
2. https://fr.foursquare.com/, visité le 04/02/2019
3. http://www.datatourisme.fr/, visité le 04/02/2019
4. https://www.wikivoyage.org/, visité le 04/02/2019
5. https://www.wikidata.org/, visité le 04/02/2019

sur le commerce électronique (Lu, Guo, 2016)(Zhang, Pang, 2015)(Minjing *et al*., 2017) mais peu courants pour la recommandation de POI.

Dans la section 2, nous présenterons les travaux connexes dans les domaines des systèmes de recommandation et de la génération d'itinéraires. La section 3 précisera les modèles et les algorithmes que nous avons élaborés dans le cadre d'une application touristique. Nous y détaillons notre démarche de recommandation de POI et de génération d'itinéraires ainsi que les premiers résultats de nos expérimentations.

2. État de l'art

Dans ce qui suit nous considérons que modèle utilisateur/touriste et profil utilisateur/touriste sont synonymes. Ainsi, un profil comprend généralement les informations qui caractérisent l'utilisateur (genre, catégorie d'âge, activités socio-professionnelles), ses préférences (préférences thématiques et historiques) et/ou son groupe d'appartenance (enfant, adulte, personne âgée). Certaines de ces informations sont fournies directement par l'utilisateur (par exemple : genre, catégorie d'âge, préférences), d'autres peuvent être déduites de ses interactions (par exemple ses « J'aime », son groupe d'appartenance). Les préférences peuvent être déduites des notes que l'utilisateur attribue aux POI ou des commentaires qu'il fait. Selon les POI que l'utilisateur a visités, il est également possible de déduire à quel groupe d'utilisateurs il appartient. Dans (Kesorn *et al*., 2017), par exemple, le profil utilisateur est caractérisé par son nom, son lieu de travail, son affiliation, son groupe d'appartenance (amis sur *Facebook*) et ses préférences sont déduites des POI que lui ou ses amis ont visités et partagés sur les réseaux sociaux (*Facebook*). Les approches proposées dans (Aliannejadi *et al*., 2016) et (Logesh *et al*., 2018), quant à elles, considèrent uniquement les préférences de l'utilisateur. Les travaux cités ici exploitent les commentaires partagés par des membres de réseaux sociaux (Yelp [6], TripAdvisor [7] et Foursquare [8]). Ils utilisent un système d'apprentissage pour déterminer, via ses commentaires, si un utilisateur a apprécié ou non une visite.

Un itinéraire est formé par une liste ordonnée de POI. L'itinéraire indique l'ordre de visite des POI et la durée estimative de visite de chaque POI et du trajet entre POI. Un système de génération d'itinéraires aide l'utilisateur à préparer son voyage. La majorité des systèmes de génération d'itinéraires se basent sur la résolution du *Orienteering Problem* (OP) qui est un système de *scoring* et trouve son origine dans la littérature de recherche opérationnelle. L'*OP* a été développé pour la première fois par Tsiligirides (Tsiligirides, 1984). Dans notre cas, l'*OP* consiste pour un touriste à collecter des points qui sont assignés à des POI. L'objectif est de maximiser le nombre de points collectés tout en respectant des contraintes (par exemple la durée de parcours de l'ensemble de l'itinéraire).

6. https://www.yelp.com/writeareview/
7. https://www.tripadvisor.fr/
8. https://fr.foursquare.com/

Plusieurs travaux étendent l'*OP*, par exemple : TOP, TDOP et TDTOPTW où T signifie *Team* - équipe (prise en compte d'une équipe, chaque POI doit être visité par, au maximum, un membre de l'équipe), TD signidie *Time-dependent* (prise en compte du temps de déplacement entre les POI), et TW signifie *Time Window* (prise en compte des horaires d'ouverture et de fermeture des visites des POI). Le processus de génération d'itinéraires est souvent composé de deux phases : le *scoring* des POI et la construction d'itinéraires. Certaines approches intègrent une troisième phase qui est l'adaptation de l'itinéraire généré. Pour cela, l'utilisateur peut intervenir pour modifier l'itinéraire (ajouter ou enlever un POI, changer l'ordre de visite de POI) et le système doit être capable de s'adapter à cette modification en respectant le contexte utilisateur comme par exemple la durée de visite (Garcia *et al*., 2010). Les approches de recommandation de POI peuvent être classées en trois catégories décrites ci-dessous : les approches basées sur le contenu, les approches basées sur le filtrage collaboratif et les approches hybrides.

Les approches basées sur le contenu utilisent des informations sur les POI et/ou sur l'utilisateur afin de faire correspondre les caractéristiques des POI aux préférences et au contexte de l'utilisateur. Plusieurs techniques peuvent être utilisées pour définir cette correspondance : les techniques d'activation-propagation (Bahramian *et al*., 2017a), les systèmes à base d'agents (Costa *et al*., 2012), les approches probabilistes (Costa *et al*., 2012), ou encore les approches inspirées de la biologie (basées sur les réseaux de neurones (Bahramian *et al*., 2017b) par exemple). De plus, (Kłopotek, 2009) considère que les systèmes de recommandation s'exposent à différents types de problèmes de démarrage à froid : les problèmes liés aux nouveaux utilisateurs (qui n'ont pas d'historique et/ou de profils), ceux liés aux nouveaux POI de la base de données (qui n'ont reçu aucune évaluation), enfin, les problèmes liés aux utilisateurs spécifiques (qui ont des préférences différentes des autres). Ainsi, les approches basées sur le contenu ne s'exposent pas aux problèmes de démarrage à froid dans le cas de l'ajout de nouveaux POI.

Les approches basées sur le filtrage collaboratif utilisent les informations concernant d'autres utilisateurs (âge, activités socio-professionnelles et/ou retours d'évaluation de visite) pour recommander des POI à un utilisateur donné. Plusieurs techniques sont utilisées par ces approches telles que les techniques de *clustering* permettant, par exemple, de grouper les utilisateurs selon leur profil (Wang *et al*., 2013) ou selon leurs liens d'amitiés sur les réseaux sociaux, ou encore, les algorithmes de type « *colonies de fourmis* » dédiés au calcul de traces de phéromones laissées au passage des différents utilisateurs (Dennouni *et al*., 2018). Dans ce dernier cas, la recommandation tient compte uniquement de la popularité d'un POI. L'avantage des approches basées sur le filtrage collaboratif est qu'elles permettent de recommander des POI sans avoir besoin d'informations précises sur ces derniers. Toutefois, le démarrage à froid est un véritable défi dans cette catégorie d'approches puisqu'il est nécessaire d'obtenir des informations relatives aux autres utilisateurs.

Les approches hybrides combinent les approches précédentes. Différentes formes de combinaison ont été envisagées (Bartolini *et al*., 2016) : (i) mise en œuvre séparée des approches basées sur le contenu et sur le filtrage collaboratif puis combinaison des résultats de recommandation, (ii) intégration de quelques traitements basés sur

le contenu dans une approche collaborative, (iii) intégration de quelques traitements basés sur une approche collaborative dans une approche basée sur le contenu, (iv) proposition d'une approche générale qui intègre à la fois des traitements basés sur le contenu et sur une approche collaborative. Dans la première catégorie, nous pouvons citer les approches de (Kesorn *et al*., 2017), (Aliannejadi *et al*., 2016) et (De Pessemier *et al*., 2015). Elles se différencient par les techniques de traitement de contenu et de prise en compte du filtrage collaboratif. Elles utilisent une fonction d'agrégation pour combiner les résultats de recommandation : (Kesorn *et al*., 2017) et (Aliannejadi *et al*., 2016) utilisent une fonction linéaire tandis que (De Pessemier *et al*., 2015) utilisent une fonction de prédiction. L'approche que nous proposons se situe dans cette quatrième catégorie des approches hybrides. A chaque itération, nous recalculons le potentiel d'un POI candidat, c'est-à-dire son score de pertinence global, en tenant compte des POI déjà intégrés dans l'itinéraire en cours de construction.

Dans ce cadre, pour mettre en valeur les POI du patrimoine pyrénéen, tout en évitant les problèmes de démarrage à froid et en nous approchant le mieux possible des préférences des utilisateurs, nous envisageons de mettre en œuvre une approche hybride qui intègre à la fois des traitements basés sur le contenu et sur une approche collaborative. Cette approche nous permettra de composer une liste de POI pertinents tout en suggérant avec parcimonie un ou plusieurs POI originaux, en complément de ceux répondant au plus près aux préférences de l'utilisateur.

3. De la valorisation à la recommandation de données patrimoniales

Afin de collecter les informations patrimoniales, les chercheurs en SHS [9] recueillent sur le terrain les informations qu'ils enregistrent par le biais de deux logiciels : RenablLP2 et Gertrude. Ces deux logiciels permettent d'exporter les données collectées au format XML. Cependant, les exports de RenabLP2 et de Gertrude sont très différents aussi bien au niveau de la structuration, de l'organisation que du nommage des informations. Pour exploiter ces données hétérogènes dans un cadre dédié au tourisme, nous avons conçu un modèle de données unifié et homogène (Bessagnet *et al*., 2018) et (Fonteles *et al*., 2018). Un premier prototype d'application fédérent ces données est actuellement disponible sur *http://tcvpyr.iutbayonne.univ-pau.fr*.

Il convient à présent de valoriser ces informations patrimoniales. Nous proposons deux types de valorisation de POI patrimoniaux : via des ressources libres de type *Open Data* (voir détails dans (Bessagnet *et al*., 2018)) et via une application mobile dont l'objectif sera de proposer des itinéraires de visite personnalisés, à destination d'un large public. Dans ce papier nous développerons davantage ce dernier point.

Une première particularité de notre approche réside dans l'utilisation de connaissances *a priori* : un thésaurus de classification de concepts patrimoniaux nous permet de catégoriser aussi bien les POI (point de vue des experts) que les préférences des utilisateurs (point de vue de l'utilisateur et/ou de l'application de recommandation). Nous disposons d'un thésaurus expert du domaine patrimonial qui comporte 4007 concepts

9. Sciences Humaines et Sociales

et 7946 liens. Ces connaissances permettent de déterminer des liens sémantiques entre POI, entre utilisateurs et entre POI et utilisateurs. Nous pouvons, par exemple, calculer la proximité sémantique entre un POI et des préférences utilisateur en mesurant leur similarité sur la base des connaissances modélisées sur ce thésaurus patrimonial. La seconde particularité de notre approche réside dans l'exploitation de traces d'interactions des utilisateurs (POI visités, notes, enchaînement de visites). Nous nous inspirons des approches de recommandation des produits commerciaux selon lesquelles « un client qui a consulté ce produit a aussi consulté ceux-ci ».

3.1. Modèles

Afin de préparer la valorisation du patrimoine via une application mobile, plusieurs modèles ont été définis.

3.1.1. Modèle Utilisateur

Le modèle utilisateur permet de décrire les éléments caractérisant les utilisateurs de notre application : genre, catégorie d'âge et préférences. Un utilisateur peut avoir plusieurs préférences thématiques (par exemple : parc, musée) et historiques (par exemple : XV^e siècle). La figure 1 illustre notre modèle utilisateur.

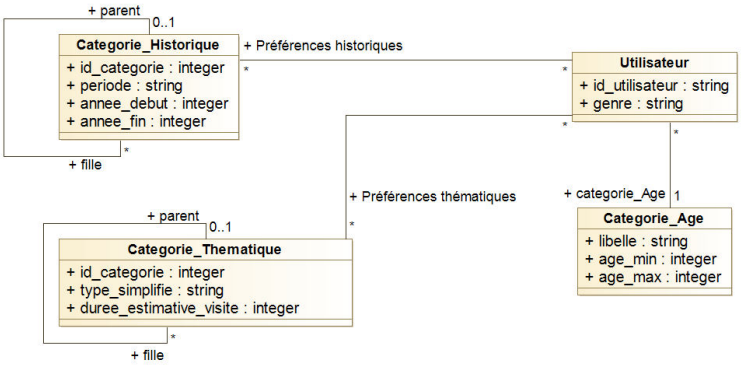


Figure 1. Modèle Utilisateur

3.1.2. Modèle de Contexte

Le contexte est composé de trois facettes : utilisateur, physique et ressource (voir figure 2). Le contexte utilisateur vient compléter le profil utilisateur avec des informations propres à une visite donnée. Ainsi, il permet de connaître le moyen de déplacement qui sera utilisé, la date, la durée et la zone géographique (ville qui doit être précisée par l'utilisateur s'il anticipe la préparation de sa visite sinon sa localisation courante). Le contexte physique décrit la localisation de l'utilisateur lors de sa visite. Cette facette change en temps réel en fonction du déplacement de l'utilisateur dans sa

zone de visite. Le contexte ressource regroupe les informations liées notamment à son appareil mobile (niveau de batterie) et à son environnement (qualité de la connectivité et de la bande passante). Ces deux dernières facettes seront utilisées au moment du parcours de l'itinéraire pour adapter ce dernier si besoin.

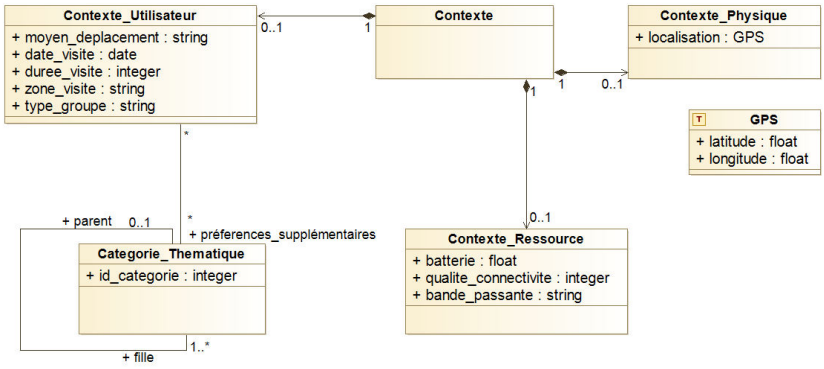


Figure 2. Modèle de Contexte

3.1.3. Modèle d'Itinéraire

Un itinéraire est une succession d'étapes dans lesquelles figurent les POI à visiter. Il est défini par sa date de génération, le moyen de déplacement utilisé et l'ensemble des étapes à enchainer pendant la visite. Chaque étape désigne un ou plusieurs POI correspondant à l'étape. De même, une étape pointe l'étape précédente et/ou la suivante. La figure 3 illustre ce modèle. Les informations sur la durée estimative de chaque déplacement et de l'ensemble de la visite sont calculées avec l'itinéraire. Tous ces modèles sont utilisés dans notre démarche de génération d'itinéraires.

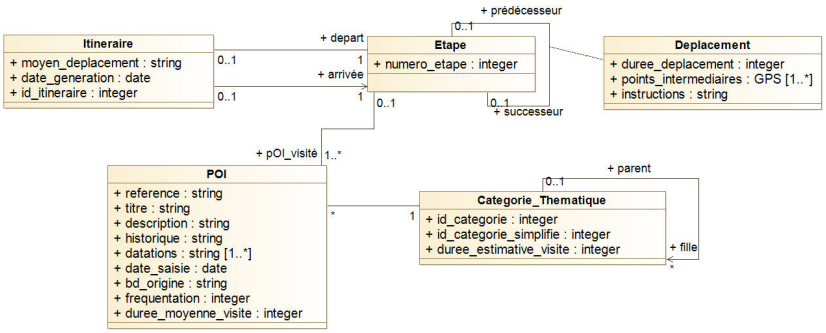


Figure 3. Modèle d'Itinéraire

3.2. Recommandation de POI pour la génération d'itinéraires

Notre approche de génération d'itinéraires fait partie des approches TDOP (voir section 2) car nous considérons le temps de déplacements entre les POI..

Algorithme 1 : Génération d'itinéraires - pseudo-code

Input : Ensemble des POI
Output : L'itinéraire composé d'une liste de POI
1 Filtre des POI de la zone (Ensemble des POI) : **Ensemble des POI candidats**
2 *Scoring* de POI (Ensemble des POI candidats) : **Ensemble des POI candidats avec leur score**
3 Sélection du POI de départ (Ensemble des POI candidats) : **POI de départ**
4 Ajout d'un POI dans l'itinéraire (POI de départ) : **Itinéraire en cours de construction**
5 **while** *(Il reste au moins un POI candidat ET Il reste du temps)* **do**
6 *Scoring* de POI (Ensemble de POI candidats restants) : **Ensemble des POI candidats avec leur score**
7 Sélection du POI suivant (Ensemble de POI candidats) : **POI suivant**
8 Ajout d'un POI dans l'itinéraire (POI suivant) : **Itinéraire en cours de construction**

La première étape de notre approche (ligne 1, algorithme 1) permet de filtrer les POI figurant dans la zone de visite de l'utilisateur. La deuxième étape supporte le *scoring* de l'ensemble des POI candidats obtenus lors du filtre (cette fonction de *scoring* est décrite dans la formule 1 de la section 3.2.1). Elle nécessite les informations relatives aux POI et les informations utilisateur (profil et contexte). La troisième étape, quant à elle, vise la sélection du POI de départ parmi cet ensemble : par défaut, le POI de score le plus élevé. Le POI de départ est ensuite ajouté dans l'itinéraire en cours de construction. Les itérations suivantes (ligne 5, algorithme 1) visent le *scoring* des POI restants suivi de l'ajout du POI de score le plus élevé dans l'itinéraire. Nous relançons le processus de *scoring* de POI candidats à chaque nouvelle itération afin d'intégrer le paramètre de temps de trajet et celui, plus original, de popularité par rapport au dernier POI de l'itinéraire en cours de construction. Cette forme de popularité prendra notamment en compte le fait que plusieurs utilisateurs qui ont visité le dernier POI ont aussi visité le POI candidat. L'itération s'arrête quand il n'y a plus de POI candidat ou bien quand il ne reste plus de temps suffisant pour poursuivre la visite. A la fin de ces étapes, nous obtenons un itinéraire qui est défini par une liste ordonnée de POI avec les durées estimatives de visite et de trajet entre les POI.

3.2.1. Entrées/sorties de données pour le scoring de POI

La génération automatique d'un itinéraire pour un utilisateur particulier nécessite le *scoring* des POI. C'est au niveau de cette fonction de *scoring* que se distinguent les différentes approches de génération d'itinéraires. Notre fonction de *scoring* (voir formule 1) correspond à la somme pondérée de trois valeurs : la pertinence du POI par rapport aux préférences de l'utilisateur ($P_{preference}$), la pertinence du POI en

terme de trajet et de durée de visite ($P_{temporelle}$) et la pertinence du POI selon le comportement des autres utilisateurs ($P_{sociale}$). Cette fonction est appelée aux étapes 2 et 6 de l'algorithme 1.

$$P(O_e,t) = \alpha * P_{preference}(O_e) + \beta * P_{temporelle}(O_e,t) + \gamma * P_{sociale}(O_e,t) \quad (1)$$

avec $\alpha + \beta + \gamma = 1$, où O_e est le POI évalué et t est le moment de calcul de *scoring*. Les valeurs des coefficients α, β et γ sont également réparties à l'initialisation et, ensuite, peuvent évoluer selon le poids que l'utilisateur souhaite donner à ses préférences, son temps de parcours ou encore à la popularité des POI.

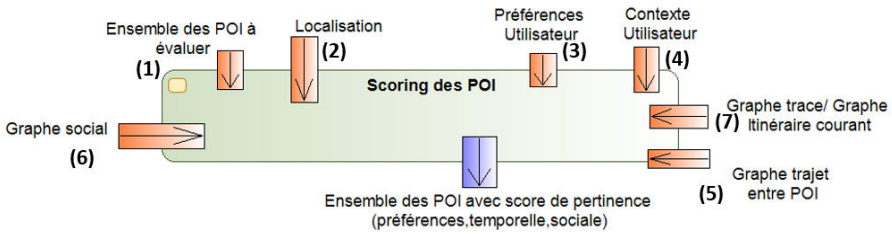
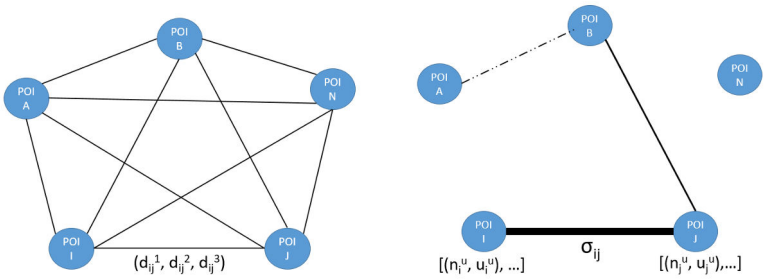


Figure 4. Données en entrée/sortie pour le scoring des POI

La fonction de *scoring* prend en entrée (voir figure 4) :

1. l'ensemble des POI à évaluer ;
2. la localisation correspondant à la position du dernier POI de l'itinéraire en cours de construction ;
3. les préférences utilisateurs composées des préférences thématiques notées $PREF_T$ et des préférences historiques notées $PREF_H$;
4. le contexte utilisateur (mode de déplacement, date, durée, zone de visite et type de groupe (avec des enfants, entre adultes ou avec des personnes âgées)) ;
5. le graphe G_{trajet} contenant les informations sur le temps de trajet entre les POI à disposition ;
6. le graphe G_{social} contenant les informations sur les évaluations de POI données par les utilisateurs ainsi que les informations relatives à leur trace de passage ;
7. la liste des POI dans l'itinéraire courant.

Le graphe G_{trajet} est défini par $G_{trajet} = < V, E >$ où V est un ensemble de POI et E est un ensemble de trajets permettant de lier deux POI. Dans notre approche, nous considérons que tous les POI sont liés deux à deux entre eux. La définition des valeurs de ce graphe est faite une seule fois sauf s'il y a des mises à jour telles qu'un ajout de POI ou des changements de durée de trajet entre certains POI. La figure 5(a) illustre le graphe de trajets entre POI. Chaque arête a un triplet de valeurs : d_{ij}^1 (durée de trajet à pied entre POI_I et POI_J), d_{ij}^2 (durée de trajet en vélo entre POI_I et POI_J) et d_{ij}^3 (durée de trajet en voiture entre POI_I et POI_J).



(a) Graphe de trajets (b) Graphe social

Figure 5. Graphes de données

Le graphe G_{social} est défini par $G_{social} = < V, E >$ où V est l'ensemble des POI et E est un ensemble de trajets réalisés entre deux POI. La figure 5(b) illustre les informations sur les traces de l'ensemble des utilisateurs du système. Chaque nœud possède une liste d'évaluations dont chaque élément est une information donnée par un utilisateur. Chaque élément est formé d'un couple de valeurs : n_i^u (note donnée par l'utilisateur u au POI_I) et u_i^u (identité de l'utilisateur u qui a visité le POI_I). Chaque arête possède une valeur qui représente l'intensité du lien entre deux POI selon les interactions des autres utilisateurs. Nous adoptons l'hypothèse que si deux POI sont fréquemment visités successivement, ils présentent un lien fort entre eux selon les utilisateurs. Pour mettre en œuvre cette hypothèse, nous nous inspirons des approches basées sur la colonie de fourmis pour lesquelles un utilisateur est représenté par une fourmi. A chaque fois qu'une fourmi passe d'un POI à un autre, elle dépose une quantité de phéromones sur son trajet. Lorsque la quantité de phéromones entre deux POI est élevée, les deux POI présentent un lien fort entre eux. Sur la figure 5(b), les phéromones sont représentées par des traits. Plus l'épaisseur de l'arête est grande, plus le taux de phéromones sur cette arête est élevé. L'avantage des approches de type « algorithmes de fourmis » est la prise en compte du nombre de passages mais également de l'éventuelle absence de passage pendant une longue durée.

Le flux de sortie obtenu à l'issue de l'exécution de la fonction de *scoring* est composé du même ensemble de POI initialement proposé en entrée. A chaque POI est désormais associé un score de pertinence.

Nous rappelons que la fonction de *scoring* est composée des trois sous-fonctions $P_{preference}$, $P_{temporelle}$ et $P_{sociale}$ (formule 1) détaillées dans la section 3.2.2.

3.2.2. Fonctions dédiées au scoring de POI

1. **Pertinence par rapport aux préférences** ($P_{preference}$)

Nous adoptons l'hypothèse que si l'utilisateur s'intéresse à une catégorie thématique ou à une catégorie de période historique alors il peut s'intéresser aux catégories sémantiquement proches de ses catégories. La fonction de pertinence en terme de pré-

férences mesure la similarité sémantique entre les catégories thématiques et/ou historiques auxquelles appartient le POI à évaluer et les catégories thématiques et/ou historiques définies comme préférences de l'utilisateur.

$$P_{preference}(O_e) = \mu * Sim_{thema}(0_e, PREF_T) + \nu * Sim_{histo}(0_e, PREF_H) \quad (2)$$

avec $\mu + \nu = 1$
Les Sim_{thema} et Sim_{histo} mesurent la similarité sémantique entre un POI et les préférences de l'utilisateur conformément à leurs caractéristiques thématique et historique respectives. Nous utilisons la mesure de similarité proposée par Wu et Palmer (Wu, Palmer, 1994) incluant la distance entre deux concepts dans un thésaurus de référence et la profondeur de leur concept parent en commun le plus proche (*LCS : Least Common Subsummers*).

2. **Pertinence temporelle ($P_{temporelle}$)**

La pertinence temporelle est définie à partir du temps de trajet ($d^i_{(O_d,\ O_e)}$) entre le dernier POI (O_d) de l'itinéraire en cours de construction et le POI candidat (O_e). Elle dépend également du temps restant pour faire la visite ($duree\_restante$). La formule 3 définit la pertinence temporelle d'un POI.

$$P_{temporelle}(O_e, t) = \frac{duree\_restante(t)\ -\ d^i_{(O_d,O_e)}}{duree\_restante(t)} \quad (3)$$

où i est le moyen de déplacement de l'utilisateur et $d^i_{(O_d,\ O_e)}$ est la durée du trajet entre O_d et O_e à l'instant t. Ainsi, le score de pertinence temporel d'un POI est d'autant plus élevé (proche de 1) que la durée du trajet estimée pour rejoindre le POI candidat est faible.

3. **Pertinence sociale ($P_{sociale}$)**

La pertinence sociale est définie à partir de l'historique, à l'instant t, du nombre de passages relevés entre le POI candidat et le dernier POI dans l'itinéraire en cours de construction. On parle d'intensité de lien entre POI. La popularité du POI candidat est également considérée dans ce calcul. La fonction d'évaluation de pertinence sociale est définie comme suit :

$$P_{sociale}(O_e, t) = \delta * popularite(O_e, t)\ +\ \phi * intensiteLien(O_d, O_e, t) \quad (4)$$

avec $\delta\ +\ \phi = 1$
où t correspond à l'horodatage du *scoring*, O_d est le dernier POI dans l'itinéraire en cours de construction, $popularite$ est la popularité de O_e et $intensiteLien$ représente le nombre de passages relevés entre O_d et O_e.

La popularité d'un POI (formule 5) est définie par la fréquentation et les retours d'évaluation (les notes) des utilisateurs. Son évaluation est faite par rapport aux popularités des POI voisins.

$$popularite(O_e, t) = \lambda * \frac{(|V|+1) - rang_{notes}(O_e, V, t)}{(|V|+1)} + \theta * \frac{(|V|+1) - rang_{freq}(O_e, V, t)}{(|V|+1)} \quad (5)$$

avec $\lambda + \theta = 1$ et où t correspond à l'horodatage du *scoring*, V est un ensemble de POI qui est formé par des POI voisins de O_e (pour des raisons de place, nous avons simplifié la formulation de V qui, en réalité doit être $V(O_e)$). Ainsi, $rang_{notes}(O_e, V, t)$ est une fonction permettant d'utiliser la moyenne des notes attribuées au POI O_e afin de déterminer, à l'instant t, son rang dans la liste des POI de V ordonnés par moyenne décroissante des notes qui leur sont associées respectivement. De la même façon, la fonction $rang_{freq}(O_e, V, t)$ permet d'exploiter la fréquentation du POI O_e et celle des POI de V afin de déterminer le rang de O_e dans la liste des POI de V ordonnés par niveau de fréquentation décroissant. Ainsi, le score de popularité d'un POI est d'autant plus élevé (proche de 1) que ses $rang_{notes}$ et $rang_{freq}$ par rapport aux voisins sont proches de 1.

L'intensité de lien entre deux POI O_i et O_j est définie comme suit :

$$intensiteLien(O_i, O_j, t) = \frac{\tau_{ij}(t)}{Q_{max}} \quad (6)$$

où t correspond à l'horodatage du *scoring*.
τ_{ij} et Q_{max} sont expliqués en détail ci-dessous.

En se référant à l'approche « colonie de fourmis », un utilisateur est désormais représenté par une fourmi. Pour cela, lorsqu'un utilisateur visite un POI O_i, puis un POI O_j, il dépose des phéromones sur le trajet entre O_i et O_j (voir section 3.2.1, figure 5(b)). A chaque passage sur le trajet O_i - O_j une quantité Q (prédéfinie) de phéromones est cumulée sur l'arête correspondante du graphe G_{social}. $\tau_{ij}(t)$ correspond à la quantité de phéromones cumulées entre O_i et O_j à l'instant t. Enfin, Q_{max} correspond à la valeur de τ maximale dans le graphe G_{social} ($Q_{max} = Max(\tau_{ij})$ où chaque τ est la quantité de phéromones cumulées sur les chemins entre les POI d'une zone géographique donnée). Q_{max} est dynamique et évolue dans le temps. Ainsi, lorsque la quantité de phéromones sur un trajet O_i - O_j est élevée (proche de Q_{max}), à cause de nombreux passages, un visiteur pourrait être invité à visiter le POI O_j lorsqu'il visite le POI O_i ou inversement.

Il est à noter que $\tau_{ij}(t)$ est stocké sur le graphe G_{social}. Cette quantité de phéromones est mise à jour à chaque nouveau passage. Ainsi, le principe de renforcement de phéromones entre O_i et O_j est défini par :

$$\tau_{ij}(t') = \tau_{ij}(t) + Q \quad (7)$$

tel que $t' > t$.
Q vaut 1 à chaque passage. De plus, la quantité de phéromones sur un trajet (notée

τ_{ij}) diminue progressivement si personne ne l'emprunte pendant un certain temps. Le principe d'évaporation de phéromones est défini par l'équation suivante :

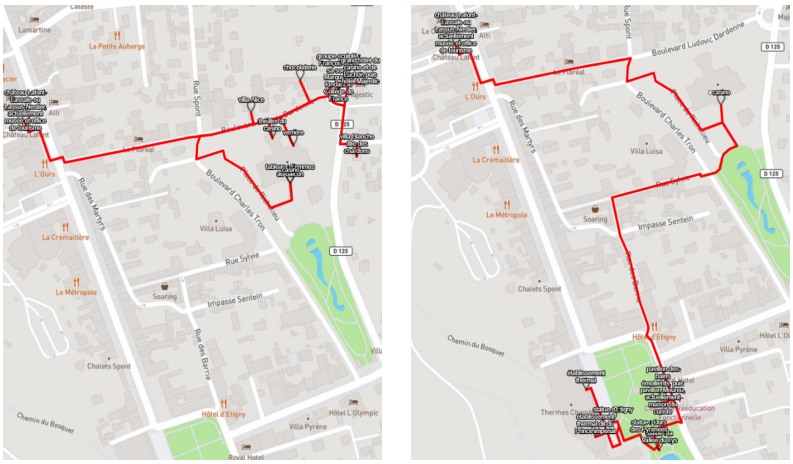
$$\tau_{ij}(t') = (1 - \rho) * \tau_{ij}(t) \quad (8)$$

où ρ est le taux d'évaporation.
Conformément aux préconisations de (Anagnostopoulos *et al*., 2017) et de (Sriphaew, Sombatsricharoen, 2015), nous faisons le choix d'une évaporation lente. Comme (Anagnostopoulos *et al*., 2017), nous fixons la valeur de ρ à 0.1 afin de garder longtemps la trace de précédentes visites tout en évitant la saturation des arcs en phéromones. Ainsi, la sérendipité est prise en compte dans le cadre de cette approche. En effet, cette gestion de l'intensité des liens consiste à proposer un nouveau POI visité par des touristes qui ont eux aussi visité le POI précédent, puis le nouveau POI, quand bien même ils n'auraient pas du tout le même profil que le touriste pour lequel on est en train de construire l'itinéraire. Ce nouveau POI ne respecte pas forcément toutes les préférences d'un touriste donné mais représente un intérêt important pour la communauté et, ainsi, permet d'étendre les centres d'intérêt du dit touriste.

3.3. Expérimentation

Nous avons mené une première série d'expérimentations selon différents scénarios et hypothèses de travail sur un jeu de données composé d'un échantillon de 120 POIs localisés à Bagnères-de-Luchon. Nous avons pour objectif d'expérimenter (i) l'algorithme de génération d'itinéraires et (ii) les pondérations α et β. Nous commentons un scénario qui n'intègre pas de contraintes sociales telles que la popularité des POIs ou l'intensité de liens entre eux (en effet, $\gamma = 0$). Dans ce scénario, nous avons défini le paramétrage suivant :
Point de départ : "place de la mairie"
Critères temporels : Temps de visite = "60 minutes" ; Mode de transport = "à pied"
Préférences : $PREF_T$ = {"musée/casino", "sculpture/peinture"} ; $PREF_H$ = {"époque contemporaine"}
Nous avons étudié deux hypothèses :
Hypothèse 1 (H1) : $\alpha = 0.5$; $\beta = 0.5$; $\gamma = 0$; $\mu = 0.5$; $\nu = 0.5$
Hypothèse 2 (H2) : $\alpha = 0.6$; $\beta = 0.4$; $\gamma = 0$; $\mu = 0.5$; $\nu = 0.5$

Nous obtenons 2 itinéraires décrits sur la figure 6. H1 (figure 6(a)) donne un itinéraire et des POIs majoritairement différents de H2 (figure 6(b)). Le choix des coefficients α et β dans le cadre de H2 resserre la contrainte de préférence - en effet les POIs proposés respectent les contraintes $P_{preference}$ avec un score moyen de 0.87 (contre 0.67 dans H1). Ce choix de coefficients relâche la contrainte de temps de déplacement court – malgré un circuit plus long, les POIs proposés respectent toujours les contraintes $P_{temporelle}$ avec un score moyen de 0.94 (contre 0.81 dans H1). Ici, nous constatons un effet inattendu avec une augmentation du nombre de POIs à visiter dû à certaines distances plus longues compensées par des temps de visite de POIs plus courts pour nombre d'entre eux. Le temps de parcours de l'itinéraire additionné au temps de visite de chaque POI respecte toujours la contrainte de temps correspondant aux critères temporels.



(a) Itinéraire H1 (b) Itinéraire H2

Figure 6. Exemple de génération d'itinéraire selon 2 hypothèses

L'algorithme propose des itinéraires composés de POIs pertinents. Ainsi, nous constatons que les POIs proposés respectent l'ensemble des contraintes : la moyenne des scores de pertinence des POIs de l'itinéraire H2 est de 0.90 tandis que celui de l'itinéraire H1 est de 0.74. H2 donne également de meilleurs résultats dans d'autres expérimentations mais demande encore à être confirmé par des tests de plus grande envergure. Les coefficients de pondération α et β jouent parfaitement leur rôle de régulation des contraintes. Après de nouvelles expérimentations, nous pourrons proposer des valeurs d'initialisation par défaut de ces paramètres.

4. Conclusion et perspectives

Nous avons présenté une architecture générale pour la fédération de données patrimoniales hétérogènes puis la valorisation de ces données. Nous avons focalisé cette présentation sur la partie recommandation d'itinéraires composés de POI patrimoniaux. Il s'agit de croiser les profils (position géographique, mode de transport, fenêtre de disponibilité, centres d'intérêts, etc.) des utilisateurs et la base de données patrimoniales pour mettre en place un système de recommandation d'itinéraires de visite comprenant des POI pertinents. Ces itinéraires sont composés de POI conformes au profil utilisateur, mais aussi d'un nombre raisonnable de POI considérés originaux du point de vue de l'utilisateur tout en méritant le détour selon la communauté. Un thésaurus de classification de concepts patrimoniaux nous permet de catégoriser aussi bien les POI que les préférences des utilisateurs. La proximité sémantique entre un POI et des préférences utilisateur peut ainsi être mesurée sur la base de ce thésaurus patrimonial. Enfin, nous intégrons l'usage d'un algorithme dit de « colonies de fourmis » afin de proposer des POI appréciés par l'ensemble des utilisateurs et qui pourront paraître originaux car ne correspondant pas exactement aux préférences de l'utilisa-

teur courant. Ce dosage subtil de POI originaux peut-être adapté via une modification dynamique des valeurs des paramètres α , β et γ de la formule 1.

L'application fédératrice est un premier prototype de visualisation des POI référencés dans la base ont été rendus publics [10]. Les travaux que nous menons offrent un accès aux données par les Systèmes d'Information Touristiques (SIT) via Wikipedia permettant ainsi aux offices de tourisme de bénéficier de ces données patrimoniales (Bessagnet *et al*., 2018).

Nous envisageons d'expérimenter cette première version de notre application de recommandation sur un panel d'utilisateurs varié et plus important. Cette expérimentation permettra notamment de définir des valeurs d'initialisation des coefficients des différentes formules de mesure de pertinence. Nous allons également comparer les résultats de notre système avec ceux issus d'une génération d'itinéraire uniquement basée sur des critères géographiques et temporels. De plus, nous travaillons sur une nouvelle version de recommandation qui intègre des éléments de contexte *physique* et *ressource* de l'utilisateur qui auront pour conséquence d'adapter dynamiquement un itinéraire. Enfin, nous envisageons d'enrichir les recommandations de notre système avec des données de l'Open Data (Datatourisme, par exemple) afin de proposer aux utilisateurs des événements culturels proches de leurs localisation et préférences.

5. Remerciements

Ce projet est réalisé dans le cadre du programme de recherche Européen TCV-PYR (2017-2020), financé par l'Union Européenne (FEDER) en partenariat avec les régions Occitanie et Nouvelle-Aquitaine.

Bibliographie

Aliannejadi M., Mele I., Crestani F. (2016). User model enrichment for venue recommendation. In *Asia information retrieval symposium*, p. 212–223.

Anagnostopoulos A., Atassi R., Becchetti L., Fazzone A., Silvestri F. (2017). Tour recommendation for groups. *Data Mining and Knowledge Discovery*, vol. 31, n° 5, p. 1157–1188.

Bahramian Z., Abbaspour R., Claramunt C. (2017b, 09). A cold start context-aware recommender system for tour planning using artificial neural network and case based reasoning. *Mobile Information Systems*, vol. 2017b, p. 1-18.

Bahramian Z., Ali Abbaspour R., Claramunt C. (2017a). A context-aware tourism recommender system based on a spreading activation method. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLII-4/W4, p. 333–339.

Bartolini I., Moscato V., Pensa R. G., Penta A., Picariello A., Sansone C. *et al*. (2016). Recommending multimedia visiting paths in cultural heritage applications. *Multimedia Tools and Applications*, vol. 75, n° 7, p. 3813–3842.

10. Code source disponible sur : https://git.univ-pau.fr/liuppa/TCVPYR/importateur

Bessagnet M.-N., Etcheverry P., Lacayrelle A., Marquesuzaà C., Rajaonarivo L., Roose P. *et al*. (2018, octobre). Leveraging heterogeneous Cultural Heritage data to promote tourism. In *Open Source Geospatial Research and Education Symposiums (OGRS).* Lugano, Switzerland. Consulté sur https://hal.archives-ouvertes.fr/hal-01976405

Costa H., Furtado B., Pires D., Macedo L., Cardoso A. (2012). Context and intention-awareness in pois recommender systems. In *6th acm conf. on recommender systems, 4th workshop on context-aware recommender systems, recsys*, vol. 12, p. 5.

Dennouni N., Peter Y., Lancieri L., Slama Z. (2018, 10). Towards an incremental recommendation of pois for mobile tourists without profiles. *International Journal of Intelligent Systems and Applications*, vol. 10, p. 42-52.

De Pessemier T., Dhondt J., Vanhecke K., Martens L. (2015). Travelwithfriends: a hybrid group recommender system for travel destinations. In *Workshop on tourism recommender systems (tours15), in conjunction with the 9th acm conference on recommender systems (recsys 2015)*, p. 51–60.

Fonteles A. S., Bessagnet M.-N., Lacayrelle A., Sallaberry C. (2018, November). Un environnement pour la valorisation de données patrimoniales hétérogènes. In *Spatial Analysis and GEOmatics (SAGEO).* Montpellier, France. (To be published)

Garcia A., Arbelaitz O., Linaza M. T., Vansteenwegen P., Souffriau W. (2010). Personalized tourist route generation. In *International conference on web engineering*, p. 486–497.

Kesorn K., Juraphanthong W., Salaiwarakul A. (2017). Personalized attraction recommendation system for tourists through check-in data. *IEEE Access*, vol. 5, p. 26703–26721.

Kłopotek M. A. (2009). Approaches to "cold-start" in recommender systems. *Studia Informatica : systems and information technology*, vol. Vol. 1(12), p. 47–54.

Logesh R., Subramaniyaswamy V., Vijayakumar V., Li X. (2018). Efficient user profiling based intelligent travel recommender system for individual and group of users. *Mobile Networks and Applications*, p. 1–16.

Lu Q., Guo F. (2016). A novel e-commerce customer continuous purchase recommendation model research based on colony clustering. *International Journal of Wireless and Mobile Computing*, vol. 11, n° 4, p. 309–317.

Minjing P., Xinglin L., Ximing L., Mingliang Z., Xianyong Z., Xiangming D. *et al*. (2017). Recognizing intentions of e-commerce consumers based on ant colony optimization simulation. *Journal of Intelligent & Fuzzy Systems*, vol. 33, n° 5, p. 2687–2697.

Sriphaew K., Sombatsricharoen K. (2015). Food tour recommendation using modified ant colony algorithm. In *5th international conference on computing and informatics, icoci 2015.*

Tsiligirides T. (1984). Heuristic methods applied to orienteering. *Journal of the Operational Research Society*, vol. 35, n° 9, p. 797–809.

Wang J., Lin K., Li J. (2013). A collaborative filtering recommendation algorithm based on user clustering and slope one scheme. In *Computer science & education (iccse), 2013 8th international conference on*, p. 1473–1476.

Wu Z., Palmer M. (1994). Verbs semantics and lexical selection. In, p. 133–138.

Zhang X., Pang X. (2015). Analysis on the mobile electronic commerce recommendation model based on the ant colony algorithm. In *2015 international conference on industrial technology and management science.* Atlantis Press.

Towards personalized Keyword Search over Relational Databases

Haithem Ghorbel [1], **Rim Faiz** [2], **Nouha Othman** [3]

1. *Université de Tunis*
   LARODEC, Tunisia
   gho.haithem@gmail.com
2. *Université de Carthage*
   LARODEC, Tunisia
   rim.faiz@ihec.rnu.tn
3. *Université de Tunis*
   LARODEC, Tunisia
   othmannouha@gmail.com

RÉSUMÉ. *La recherche par mots-clés dans les bases de données relationnelles a suscité un intéet grandissant ces dernières annèes en raison de sa facilité d'utilisation. La plupart de ces recherches non seulement nécessitent un accès préalable aux données, ce qui restreint leur applicabilité si cette condition n'est pas vérifiée, mais aussi renvoient des réponses très génériques. Cependant, fournir aux utilisateurs des réponses personnalisées est devenu plus que jamais nécessaire en raison de la surabondance de données qui peut déranger l'utilisateur. Inspiré par l'application réussie de la technique de filtrage collaboratif dans les systèmes de recommandation, nous proposons une nouvelle approche basée sur les mots-clés pour fournir aux utilisateurs des résultats personnalisés basés sur l'hypothèse que seulement une information sur le schéma de la base de données est fournie.*

ABSTRACT. *Keyword search in relational databases has attracted wide attention over the last years owing to its ease of use. Most recent research not only requires a prior access to the data, which severely restricts their applicability if this condition is not verified, but also returns generic answers. Nevertheless, giving users personalized answers has become more than ever necessary due to the overabundance of data which can be annoying for the user. Inspired by the successful application of the collaborative filtering technique in recommender systems, we propose in this paper a new approach to provide users with customized results based on the hypothesis that only information on the database schema is provided.*

MOTS-CLÉS : *Recherche par mots-clés, systémes de recommandation, Bases de données relationnelles.*

KEYWORDS: *Keyword search, Recommender systems, Relational databases.*

1. Introduction

Over the decades, an explosive amount of structured data has been stored side by side in Relational Databases (RDB). RDB have been widely used owing to the rich information they provide including links between the different entities in the DB. Developing effective query methods for users to easily query huge and complex repositories without the need of technical expertise has become one of the biggest challenges of the database community (Agrawal *et al*., 2002 ; Aditya *et al*., 2002).

In fact, the emergence of web search engines has made keyword search the most commonly used search technique. The strength of this latter is that it enables users to easily express their information needs by a few keywords without the need to have knowledge of the database schemas or structured query languages. Nevertheless, such an approach requires a prior access to the data instance in order to build the indices that will pinpoint the different tuples associated with the keywords at run time (Bergamaschi, Domnori *et al*., 2011). This is a considerable shortcoming since it limits their applicability if a prior access to the data is not possible. Another significant limitation is that the inter-dependencies between the query keywords were ignored. In fact, in reality, a keyword in a query depends on both the relationship between the keyword and the data structure, and the data to which the other keywords in the query are modeled. More concretely, the meaning of each keyword in a user's query also depends on the meaning of the others. On the other hand, with the tremendous development of information technology, the amount of data has been growing exponentially. Thus, finding the desired information in a massive database has become a crucial but also a challenging task. As a matter of fact, the user has to select the appropriate keywords and complete the task of search systems to find the desired answer, which is a tedious task and requires not only an additional effort but also a considerable loss of time. Recommendation Systems are actually a powerful tool to filter data, providing only what the user is most likely looking for. Recently, RSs have been widely used especially in e-commerce websites which take advantage of the personalization ability of these systems to better predict user's intent.

In the present paper, we propose, to the best of our knowledge, the first successful attempt that combine RSs techniques and RDB to overcome the limits of the keyword search in the underexploited context: Keyword search over RDB when no prior access to data is possible. Our novel approach named DBRec aims at returning personalized answers when we have no prior access to DB. The remainder of the paper is structured as follows. Section 1 gives an essential background on keyword search over RDBs and RSs respectively. In Section 2, we review the main existing work on querying RDBs as well as the main methods of RSs. Then, we describe our proposed approach in Section 3. Subsequently, we present in Section 4 our experimental evaluation done to assess the effectiveness of our approach and we report our findings. Towards the end, we conclude and outline our challenges and perspectives.

2. Related work on Querying Relational Databases

Arguably, keyword search has become the standard for seeking information on the Web as it allows the user to easily formulate queries with a few keywords. However, its

simplicity comes with a price; keywords are fraught with ambiguity and their intended meaning needs to be explored further (Wang *et al*., 2008).

Over the years, the Information Retrieval (IR) community has developed advanced approaches for keyword search over documents to return relevant answers to the user. These approaches, though, don't return good results with RDB systems (RDBs), as IR-style search considers tuples as unstructured data, while in RDBs, the retrieved information is spread among tables. Unlike textual documents, the tuples are linked through the foreign-primary key constraints. Thus, foreign-primary key paths connecting tuples that contain keywords, represent an essential ingredient for solving a keyword query over a database. Defining representation models for databases to retrieve these paths is crucial. For these reasons, the direct application of keyword-based approaches to relational databases, where information is fragmented in numerous tables, is neither efficient nor effective (Bergamaschi *et al*., 2014). Indeed, multiple systems were proposed in the literature, where the most popular ones are BANKS (Aditya *et al*., 2002), BANKSII (Kacholia *et al*., 2005), DBXplorer (Agrawal *et al*., 2002), DISCOVER (Hristidis, Papakonstantinou, 2002), and SQAK (Tata, Lohman, 2008) and the most recent one are KEYMANTIC (Bergamaschi, Domnori *et al*., 2011), KEYRY (Bergamaschi, Guerra *et al*., 2011) and SEMINDEX (Chbeir *et al*., 2014). The objective of these systems is to better cover a keyword query, in order to return answers that matches the user's intent. These approaches can be classified into two broad categories: schema based and tuple-based approaches. The Schema-based approaches model the database schema as a graph, in which the nodes express database relations and edges express interdependence between primary and foreign keys. In these approaches, two phases are needed to build the query answer. First, possible paths are developed from relations that contain a query keyword and the schema graph. Second, fitting queries related to the current tuple are developed following the generated paths. Such approaches can fulfill a keyword query by the use of the schema information to generate SQL queries in RDBs, such as in DBXPLORER, DISCOVER, PRECIS, SQAK, KEYMANTIC, and KEYRY systems. Tuple-based approaches such as BANKS and BANKS II, model the database as a data graph, wherein nodes represent the tuples and edges denote the relationships between a pair of tuples, such as foreign key or primary key dependencies. Unlike schema-based approaches, only one phase is involved for the answer generation, where the schema extraction and the tuple retrieval tasks are mixed. The particularity of a data graph is that nodes and edges are typically weighted, which provides users with more information on how the objects are interconnected.

KEYMANTIC, KEYRY and QUEST (Bergamaschi *et al*., 2016) tackled the issue of keyword search over RDBs differently; they can provide answers to the user's query without the necessity of a prior access to the data stored in the database to build indices that will locate the tuples. These techniques use intentional knowledge. In other words, they provide results even if we know only the schema of the database. Therefore, KEYMANTIC and KEYRY consider the inter-dependencies of the keywords in the query to compute the best answers. KEYMANTIC computes intrinsic and contextual weights to consider inter-dependencies between keywords and the mappings. For the generation of the top-k best mappings, an extension of the Hungarian algorithm is used. KEYRY is based on the Hidden Markov Models for the generation of the pos-

sible configurations, and generalizes the Viterbi algorithm finding the top-K maximum likelihood state sequences.

3. Related work on Recommender Systems

Nowadays, the plethora of available data makes filtering data more than ever necessary to improve the quality of data. In this direction, Recommender Systems (RSs) are designed to personalize users' search, and provide only what might interest them even if they cannot formulate good queries. RSs have been widely used in recent years as they are able to guide users in a personalized way to interesting objects in a large space of possible options (Ricci *et al*., 2011). Indeed, these systems can predict user's needs and even surprise the user with recommendations that he doesn't know they exist. Basically, RSs can be categorized into three groups: Collaborative Filtering, Content-Based and hybrid approaches.

Collaborative Filtering is probably the most popular and widely used technique in RSs (Ricci *et al*., 2011). Basically , it recommended information to a user according to the history valuation of all users communally. The main drawback of this technique is the cold-start problem. That is to say, CF based systems may have difficulties to provide recommendations with new items and new users, when no sufficient ratings for items or no enough users are available in the system. Content-Based (CB) recommenders also called Information Filtering, relies on item's description or uses content in the historical search, to guess the potential items that the user might be concerned with. This type of recommender systems, based on content of items rather than on other users' opinions or interactions, use the similarities of the attributes between items. For each item, a profile containing all properties must be created.

A content-based RS was proposed by (Amatriain, Mobasher, 2014) based only on the profile built up by analyzing the content of items rated by the users in the past. The recommendation process basically consists on matching up the attributes of the user profile with the attributes of a content object. Generally, these systems encompass three major components (Ricci *et al*., 2011): the Content Analyzer which describes the items' content, the Profile Learner which collects data representative of the user preferences and tries to generalize this data in order to construct the user profile, and the Filtering Component which utilizes the user profile to propose apposite items by matching the profile representation against that of items to be recommended. Unlike the CF technique, CB does not suffer from the item cold-start problem, i.e. it has no problem with recommending new and unpopular items. However, the main shortcoming of this technique is that the content analysis often requires domain knowledge and sometimes it is not easy to extract from the items enough relevant information to describe them. Another limitation is that serendipity is hard to obtain with this model. Furthermore, CB recommenders suffer from the overfitting problem when there are not large enough ratings in the training stage. It is worth mentioning that there have been several hybrid recommender systems that combine both CF and CB models to benefit from the characteristics of both and scale down their limitations. For instance, CF systems are more domain-independent than CB systems, while CB can handle better the cold start problem for both new users and new items. However, the aforementioned approaches require a prior access to data to build indices that will

locate the tuples related to the given keywords, at run time, except for KEYRY and KEYMANTIC. The necessity of a prior access to data is not fitting reality, since in some cases we cannot access the data contents of a DB such as in e-commerce sites which are subject to recurring updates. To the best of our knowledge, only KEYRY and KEYMANTIC attempt to resolve this issue. KEYRY is based on the Hidden Markov Model, while KEYMANTIC uses essentially multiple similarity measures with the help of some external knowledge. Both techniques mainly exploit terms of the query and the schema information of the DB. Nevertheless, these works are too static since they rely only on similarity distances and some external knowledge which leads to very generic answers and specifically not user-centric responses. In fact, users can query the same information need differently or also may write the same query for different information needs. Obviously, understanding the user preferences is crucial to better address its query and identify what he is really expecting. Thus, we propose a novel approach to overcome the limitation of the keyword search when no prior access to data is possible and deliver personalized and relevant answers to the user's query.

4. Proposed Solution: DBRec

The core idea of our approach, called *DBRec*, is to combine the keyword search over RDB with some techniques used in recommender systems. DBRec is illustrated in Figure 1

and aims at integrating some recommendation and databases concepts to get better personalized answers to a simple keyword-based query posted by a user. It provides recommendations and serendipitous answers even when no prior access to the database is allowed, relying on both schema and users information. The components of the given approach are detailed below.

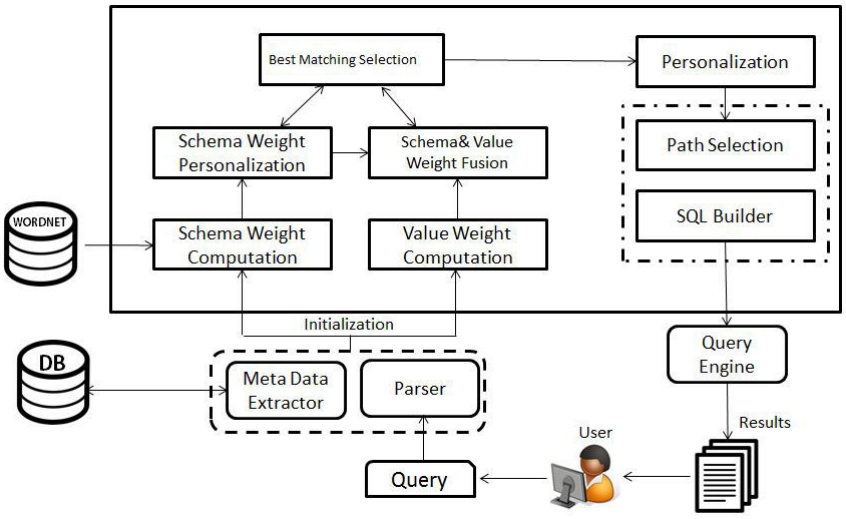


Figure 1. Global architecture of the DBRec Approach

4.1. Schema Terms Matching

Basically, the initialization step includes two phases namely, Schema Weight Computation and Schema Weight Personalization. The meta data extractor extracts the schema information provided with the database to load the first line of the Schema Weight (SW) matrix. The parser separates each term of the query whereby we fill the first column of the given matrix.

4.1.1. Schema Weight Computation

This phase consists in determining which keywords match with schema terms (attributes, relations) starting from a query and the schema information of the database. Attributes and relations are considered as metadata. Different string similarity measures can be used to estimate the keyword-attribute/relation distance such as Levenshtein, Hamming and Cosine distance. Note that Hamming is defined for strings of equal length which is not adequate for our context. We opted for the Levenshtein measure which computes the minimal number of insertions, deletions and replacements needed for transforming a string X into a string Y. However, a simple string similarity between the keyword and the schema term is not enough due to the heterogeneity of the user's vocabulary. For this purpose, we employ WordNet for Word Sense Disambiguation (WSD) to obtain the synonyms, hyponyms and hypernyms related to a given word. Due to the wide available vocabulary, a user may use different words that do not appear in the schema information of the database. Each used keyword is compared to all the synonyms, hyponyms and hypernyms of every schema term to keep the one having the highest similarity.

4.1.2. Schema Weight Personalization

This phase consists in updating the Schema Weight Matrix by making use of the information gathered from the users' profiles. The main idea here is to add the concept of collaborative filtering of the RSs; build profiles for users and use their search history as well as similar users' history to personalize results.

We compute the Personalized Schema Weight (PSW) that uses sessions information to update the SW matrix. Similar sessions are indexed in a table T. Then, we calculate how many times every schema term had the maximum value, for all the queries in the similar sessions. We store the result of this computation in its specific column in the SW matrix. Each value of this column is combined with each one in its analogue column in the SW to get the new PSW values in their corresponding cells. In other words, we weight the values of the SW of each column by a variable that affects the first values depending on the number of times this column had the maximum value.

4.1.3. Best Matching Selection

Computing the best possible matching of keywords to database terms is known as the assignment problem. In fact, we have a number of keywords and a number of schema terms. Any keyword can be assigned to any schema term, incurring some cost that may vary depending on the keyword-schema term assignment. It is required to match all keywords by assigning exactly one keyword to each schema term and

exactly one schema term to each keyword in such a way that the total weight of the assignment is maximized. The popular Munkres, a.k.a. Hungarian, algorithm (Munkres, 1957), is a possible solution to this problem but, it provides only the best matching. (Bergamaschi, Domnori *et al*., 2011) adapted this algorithm to our context, to not stop after the generation of the top one mapping, but continue to generate the other best ones. Besides, the weight matrix is dynamically updated every time a mapping of a keyword to a database term is decided during the calculation.

4.2. Value Weight Contextualization

Value Weight Contextualization includes the value weight computation and the schema value weight fusion phases described below.

4.2.1. Value Weight computation

The Value Weight (VW) matrix computation is performed in exactly the same way as in (Bergamaschi, Domnori *et al*., 2011). The computation is mainly done within the domain information of attributes. KEYMANTIC employed a semantic distance to estimate the relatedness of two concepts. Therefore, every matrix cell in the VW contains a value as an indicator of the eligibility and suitability of the keyword with the attribute domain. The keywords that have already been mapped to schema in the previous step will get assigned 0 in every cell of their lines to ensure that they won't be recomputed in the VW.

4.2.2. Schema Value Weight Fusion

After the computation of the best mapping to schema terms M_i and the value weight matrix, the value weight matrix is updated according the the terms mapped to schema. In keywords queries, a keyword may refer either to a schema term or to a value in a schema term. We contextualize the value weight matrix according to terms mapped as schema terms from the PSW, taking into consideration the keywords positions in the query. The user can use more than one term to describe one concept. We propose the Algorithm 4.2.2 to deal with this issue. This algorithm enables to check for each keyword k if it corresponds to any schema term x from the mapped ones. Then, if k corresponds to a keyword mapped to a schema term x, two situations may arise: If x is a relation R, we add a weight Ω to all the attributes of R for every adjacent keywords A(k) $\cup$ B(k), otherwise, if x is an attribute A of a relation R, we increase the weights of this attribute and the related attributes (with functional dependencies) by Ω for all A(k) $\cup$ B(k) keywords neighboring k.
A(k) and B(k) are two functions that retrieve the following and preceding keywords neighboring k respectively. Ω is a variable, its value is proportional to the distance between keywords. The output of this step is a contextualized Value Weight Matrix $V_j(M_i)$. Again, we will use the extension of the Hungarian Algorithm, this time over $V_j(M_i)$, to get the best assignments.

Algorithm 1 Updated Value Weight Matrix

Input: VW: Value Weight Matrix, Q : Query, M_i : Mapping of the SW
Output: Updated Value Weight Matrix
Compute SVW(VW, Q, M_i) **begin**
  foreach $w \in Q$ **do**
    $x : \exists(w, s) \in M_i$
    if *x = null* **then**
      Continue to the next w
    end
    if *x is a Relation R* **then**
      foreach *attribute A of R* **do**
        foreach *w' ∈ A(w) ∪ B(w)* **do**
          SVW[w', R.A] ← VW[w', R.A] + Ω
        end
      end
    end
    if *x is an attribute A of a relation R* **then**
      foreach *w' ∈ A(w) ∪ B(w)* **do**
        SVW[w', R.A] ← VW[w', R.A] + Ω
      end
      foreach *attribute A' of R' ∃ join path from A to A'* **do**
        SVW[w', R'.A'] ← VW[w', R.A] + Ω
      end
    end
  end
  return SVW
end

4.3. Generation of the personalized query answers:

The $V_j(M_i)$ with its related M_i is a full matching of the keywords to DB terms, producing together a new combination named A_ij. The score of each combination is the sum of weights of the $V_j(M_i)$ and its associated M_i.

SQL queries can be achieved with the possession of the first-score combinations. Yet, this latter just reside on the keywords match to their adequate database terms, without defining the relations between the terms. Works that process under the same assumption of no prior access to data such as (Bergamaschi, Domnori *et al.*, 2011) only consider the similarity between keywords-attributes/domain of attributes' similarities, which is not always appropriate. To cope with this limitation, we take advantage of the profiles built by our approach, to personalize the answers. Giving a current user making a query in his current session, we compute the similarity of this query with all previous queries in all the sessions using the cosine similarity. The answer that gained interest of the user is credited even if its associated query is fairly similar to the user's current query.

In a nutshell, our contribution consists in adding a Collaborative Filtering aspect to a previous proposed solution for querying databases without a prior access to data. Based on generated profiles, which are in our case a set of sessions created by users, we guarantee a more personalized answer. We try to take into account users' interactions with the system to better predict user's expected answers. The proposed approach ensures more serendipity thanks to the CF that can offer a controllable serendipity.

5. Experimental Evaluation

5.1. Experimental Setup

In our experiments, we used MySQL 5 as the relational database management system and WordNet 3.0 [1] as a lexical database. We explained the content of the Database to no technical users without exposing its schema information. Then, we asked them to propose keywords queries and describe what they expect as answers for their queries. Thereafter, an expert formulates an SQL query for this purpose. We compared the results generated by our approach with those obtained by the expert. We used a fraction of the MovieLens [2] 100K database. In our tests, 18 users were involved, we group each user's queries in a single session. The number of sessions is between 1 and 3 for each user. We used 105 queries distributed among the users sessions. To evaluate DBRec, we were mainly based on the following parameters:

– Number of keywords: As keywords are the main information we are analyzing in our approach, we varied the number of keywords to see its impact on the results.

– Similarity threshold: It principally defines the number of the possible answers. We ranged it from 0.6 to 0.9 and we compare the results with those of KEYMANTIC.

– Number of sessions: As we were based on the Collaborative Filtering aspect, it makes sense to study the impact of sessions on the approach.

Note that for the initialization, we used the Information_schema views provided by MySQL which allows to retrieve metadata about objects in the DB.

5.2. Experimental Results

Figure 2 shows a comparison between KEYMANTIC and DBRec in terms of the percentage of the *1st position answers* which represents the percentage of expected answers by the user generated as the first answers by the system, the percentage of the *not in first position answers* which denotes the expected answers which are not in first positions and the percentage of the *not found answers* which denotes the relevant answers that the system fails to return.

Experiments showed an amelioration in the percentage of the *1st position answers* over the *not in first position answers*. However, the percentages of the *not found answers* for both approaches are quite similar. We run DBRec every time we add a

1. www.wordnet.princeton.edu
2. www.MovieLens.com

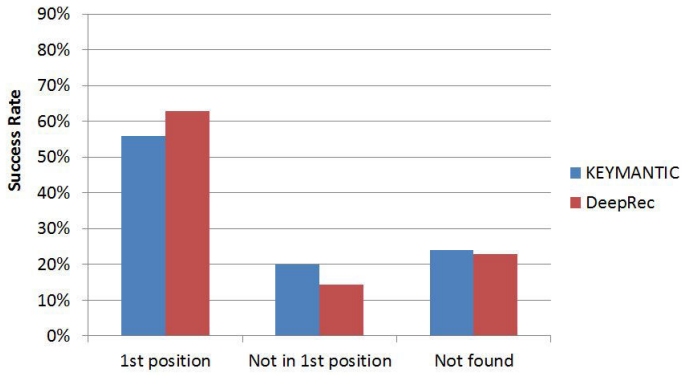


Figure 2. Generated Success Rate for DBRec and KEYMANTIC

session. For the very first sessions, we obtain nearly the same results as KEYMANTIC. This means that even with a cold start, results quality does not decrease. Yet, sessions after sessions some answers have changed, until we obtain the results given in Figure 2. The results of success rate according to the number of sessions are given in Figure 3.

For the keywords number and its impact on the success rate, we computed the percentage of answers that were considered as relevant to the users and returned as the first answer while changing the keywords number in the query as shown in Figure 3.

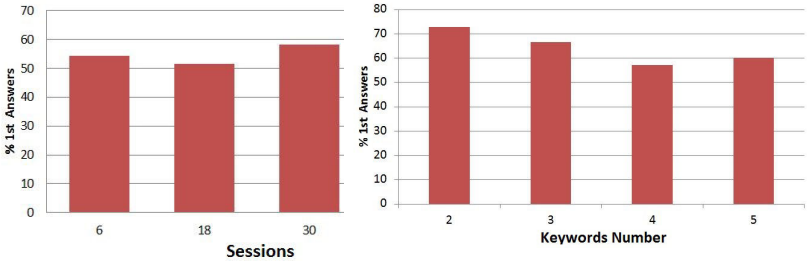


Figure 3. Generated Success Rate according to Sessions and Keywords Number

We remarked that no changes have been noticed for the *not found answers*. Some queries can generate the expected answer, but mostly, not in the first position, except for the ones that have been 'liked' before by the user. The processing time varies from few seconds to minutes, depending on the number of tables related to the queries. For example, the queries that concern one table take around 2 seconds. However, the response time can reach around 2 minutes for the queries including several tables. The execution time needed for both approaches to generate answers is quite similar. The similarity threshold is used to determine the minimum similarity distance required between keywords and terms to be retrieved from the DB. Figure 4 shows how the system reacts when varying the similarity threshold. We can observe the descending

allure of this histogram, showing that the larger the value of the threshold is, the less the analyzed terms of DB are and the less the execution time is.

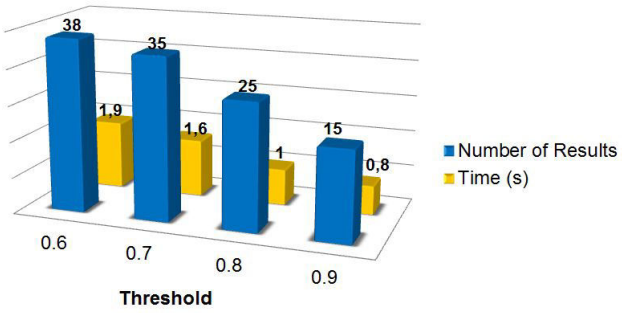


Figure 4. Effect of varying the threshold on DBRec results

5.3. Discussion

Interesting results are presented for users' interactions including sessions and queries. In fact, the number of answers responding to users' expectations has increased. As any collaborative filtering based system, the more users we have and the more interactions they make with the system, the better its performance is. Experiments show that there is no optimal similarity threshold, it depends on the query and users' expectations. By increasing the threshold, both results number and time are reduced. However, by decreasing it, better results are obtained, with a higher serendipity, but it comes with the price of time. Moreover, there isn't an optimal number of keywords. The more words are contained in the query, the longer the time required for results generation. However, the number of keywords may increase either accuracy or serendipity depending on the terms employed by the user. The cold start problem of the Collaborative Filtering technique is resolved by DBRec. This latter allows to have results even with new users, thanks to a back strategy step, insured by Personalized Schema Weight, which maintains the values of the initial matrix obtained before personalization. Furthermore, the personalization step allows to take into account every result that fits the user's request. This is very useful when a user searches an information that has previously sought. However, with KEYMANTIC, regardless the number of times the user interacted with the system, it always recomputes the answers ignoring what the user is probably expecting. The schema quality is a crucial parameter for DBRec. Indeed, having a database containing tables with nearly the same names, structure and attributes data-types may have some limitations. A considerable one is the time needed to generate answers, few minutes for a single query is relatively long, but it is still better than having no result.

6. Conclusion

This paper addressed the problem of processing keyword queries over relational databases under the assumption that no prior access to data is possible. Our contribu-

tion consists on providing users with personalized results in this specific context. by extending existing approach with new components and resources. These new components are the personalization of the schema weight matrix and the answers respectively, while the resources are users and information concerning their interactions with the systems. Beyond simply returning generic answers, our findings indicated that our approach called DBRec provides personalized results that better fit the users' intent. Before the generation of the answers, we took advantage of the current user profile to further personalize answers based on the CB technique of RSs; favoring an answer already liked by the user if the similarity between queries are higher than a given threshold. Providing personalized answers when no prior access to data is possible, makes DBRec usable in a large range of applications. In the future, it would be interesting to integrating a semantic-based feature in our approach in order to decrease the percentage of the not found answers.

Bibliography

Aditya B., Bhalotia G., Chakrabarti S., Hulgeri A., Nakhe C., Parag P. *et al*. (2002). Banks: Browsing and keyword searching in relational databases. In *Proceedings of the 28th international conference on very large data bases*, p. 1083–1086.

Agrawal S., Chaudhuri S., Das G. (2002). Dbxplorer: A system for keyword-based search over relational databases. In *Data engineering, 2002. proceedings. 18th international conference on*, p. 5–16.

Amatriain X., Mobasher B. (2014). The recommender problem revisited: Morning tutorial. In *Proceedings of the 20th ACM SIGKDD international conference on knowledge discovery and data mining*, p. 1971–1971.

Bergamaschi S., Domnori E., Guerra F., Trillo Lado R., Velegrakis Y. (2011). Keyword search over relational databases: A metadata approach. In *Proceedings of the 2011 acm sigmod international conference on management of data*, p. 565–576. New York, NY, USA.

Bergamaschi S., Guerra F., Interlandi M., Trillo-Lado R., Velegrakis Y. (2016). Combining user and database perspective for solving keyword queries over relational databases. *Information Systems*, vol. 55, p. 1–19.

Bergamaschi S., Guerra F., Rota S., Velegrakis Y. (2011). A hidden markov model approach to keyword-based search over relational databases. In *Conceptual modeling er 2011*, vol. 6998, p. 411-420.

Bergamaschi S., Guerra F., Simonini G. (2014). Keyword search over relational databases: Issues, approaches and open challenges. In *Bridging between information retrieval and databases*, vol. 8173, p. 54-73.

Chbeir R., Luo Y., Tekli J., Yetongnon K., Ibanez C. R., Traina A. J. *et al*. (2014). Semindex: Semantic-aware inverted index. In *Advances in databases and information systems*, p. 290–307.

Hristidis V., Papakonstantinou Y. (2002). Discover: Keyword search in relational databases. In *Proceedings of the 28th international conference on very large data bases*, p. 670–681.

Kacholia V., Pandit S., Chakrabarti S., Sudarshan S., Desai R., Karambelkar H. (2005). Bidirectional expansion for keyword search on graph databases. In *Proceedings of the 31st international conference on very large data bases*, p. 505–516.

Munkres J. (1957). Algorithms for the assignment and transportation problems. *Journal of the Society of Industrial and Applied Mathematics*, vol. 5, nº 1, p. 32–38.

Ricci F., Rokach L., Shapira B., Kantor P. B. (2011). *Recommender systems handbook* (vol. 1).

Tata S., Lohman G. M. (2008). Sqak: doing more with keywords. In *Proceedings of the 2008 ACM SIGMOD international conference on management of data*, p. 889–902.

Wang H., Zhang K., Liu Q., Tran T., Yu Y. (2008). Q2semantic: A lightweight keyword interface to semantic search. In *The semantic web: Research and applications*, vol. 5021, p. 584-598.

Conception et formalisation d'un système de recommandation de prestations touristiques

Olivier Hotel [1], **Frédéric Pourraz** [1], **Hervé Verjus** [1]

Université Savoie Mont-Blanc
LISTIC
Annecy, France
{olivier.hotel,frederic.pourraz,herve.verjus}@univ-smb.fr

RÉSUMÉ. *Dans cet article, nous introduisons la notion d'avatar pour les systèmes de recommandation. Cette notion permet non seulement d'apporter une solution au problème du démarrage à froid mais aussi de manipuler des informations incomplètes concernant les utilisateurs du système de recommandation. Nous décrivons également un processus en trois étapes permettant d'affiner ces informations : identification des avatars, partitionnement des utilisateurs et application de la méthode des facteurs latents. Enfin, nous décrivons un exemple de notre méthodologie dans le cadre d'un système de recommandation de prestations touristiques.*

ABSTRACT. *In this article, we introduce the notion of avatar for recommender systems to solve the cold start problem and to manipulate incomplete information about the users of the recommender system. We also describe a three-step process for refining this information: avatars identification, clustering of the users, latent factors method. Finally, we describe a concrete example of our metodology as part of a recommender system of tourist services.*

MOTS-CLÉS : *Système de Recommandation - Informations Incomplètes - Système d'aide à la Décision - Avatar - Partitionnement - Facteurs Latents*

KEYWORDS: *Recommender System - Incomplete Information - Business Intelligence - Avatar - Clustering - Latent Factors*

. – /

1. Introduction

De nos jours et avec le volume croissant de données disponibles, la conception de systèmes de recommandation est un axe de recherche important. Ces systèmes sont largement utilisés dans les plateformes de commerce en ligne et tendent à se développer dans de nombreux secteurs. Sans remplacer l'humain, un système de recommandation l'aide dans ses choix en lui faisant des suggestions. Il appartient à l'humain, de l'utilisateur novice à l'expert aguerri, de suivre les propositions qui lui sont faites par le système de recommandation, de s'inspirer de ces propositions tout en les accommodant ou, de ne pas en tenir compte.

Cet article propose un cadre générique et indépendant d'un domaine métier spécifique pour concevoir un système de recommandation couvrant les spécificités suivantes: au regard d'un ensemble d'articles, d'utilisateurs et de notes attribuées aux articles par les utilisateurs, le système de recommandation sera en capacité de faire des propositions pertinentes à l'utilisateur, qu'il ait ou non connaissance au préalable de ce dernier. Pour cela, et après avoir présenté un état de l'art des méthodes de recommandation et des problématiques qui s'y rapportent en section 2, nous introduisons une notion d'avatar au sein du système de recommandation en section 3; et nous décrivons des mécanismes d'adaptation en section 4.

2. État de l'art

Sans prétendre à une totale exhaustivité, cette section s'efforce de présenter succinctement un état de l'art des méthodes de recommandation proposées dans la littérature s'inscrivant dans le cadre précédemment décrit. Une description plus détaillée est disponible dans (Béchet, 2016 ; Benouaret, 2017). Ces méthodes peuvent être divisées en trois familles:

1. les méthodes basées sur le contenu (Pazzani, Billsus, 2007) consistent essentiellement à comparer les attributs des objets avec les attributs d'intérêt de l'utilisateur actif;

2. les méthodes basées sur des principes de filtrage collaboratif (Schafer *et al.*, 2007) reposent sur la conjecture que les opinions des autres utilisateurs peuvent être utilisées pour prédire les préférences des autres utilisateurs, et sur l'hypothèse que si des utilisateurs ont les mêmes préférences sur certains objets alors ils auront également les mêmes préférences sur d'autres objets; et

3. les méthodes hybrides qui combinent les deux précédentes approches de recommandation.

Ces dernières années, des propositions ont émergé (Chen, Chen, 2015 ; Hernández-Rubio *et al.*, 2018) afin de cerner le profil de l'utilisateur à partir d'informations recueillies sur Internet, des traces numériques que l'utilisateur aurait pu laisser (réseaux sociaux, forums, blogs,...); ces approches tentent d'éviter le problème du démarrage à froid en faisant l'hypothèse que des informations permettant de qualifier l'utilisateur existent et sont disponibles.

Bien que les approches décrites dans cette section permettent, dans la plupart des cas, de recommander des objets pertinents aux utilisateurs, elles sont inopérantes lorsqu'aucune information concernant les objets ou les utilisateurs n'est disponible. Ce problème de démarrage à froid est encore largement ouvert même si des approches permettant d'ajouter un nouvel utilisateur ou un nouvel objet au système de recommandation ont été proposées dans la littérature (Negre *et al*., 2013).

3. Proposition d'avatars

3.1. Définition

Dans le cadre de notre approche, nous introduisons la notion d'avatars pour résoudre le problème du démarrage à froid. Un avatar est une représentation fictive choisie par l'utilisateur pour représenter ces intérêts.

Dans la phase initiale de fonctionnement du système, l'identification des avatars doit répondre à un triple objectif :

1. les utilisateurs doivent pouvoir s'identifier ne serait-ce que partiellement à un des avatars proposés par le système;

2. il doit être possible d'associer des notes fictives à chaque couple $objet - avatar$; et

3. le nombre d'avatars doit être limité pour que chaque utilisateur puisse rapidement choisir son avatar.

3.2. Étude de cas

Dans le cadre du projet Européen Interreg France-Suisse Transfrontour dont un des objectifs est la conception d'un système de recommandation de prestations touristiques, des études et des sondages réalisés auprès des acteurs du tourisme du département de la Haute-Savoie (74) ont permis d'identifier les avatars des usagers des différents services touristiques : *tourisme d'affaires*, *tourisme familial*, *tourisme culturel*, *tourisme gastronomique*, *tourisme sportif* et *tourisme d'hiver*.

Ces mêmes études ont également permis d'identifier les différentes prestations touristiques, de les classer par grands domaines (*hébergement*, *restauration*, *activités culturelles*, *activités sportives*, etc.) et par sous-domaines (restauration : *gastronomie* - *snack*; activités culturelles : *patrimoine* - *art*; activités sportives : *sport outdoor* - *sport indoor*, etc.).

Dans l'objectif de pouvoir associer une note à chaque couple $avatar-prestation$, nous proposons d'utiliser les sous-domaines identifiés comme des descripteurs permettant de représenter formellement les avatars des usagers ainsi que les différentes prestations touristiques et nous associons à chaque descripteur un poids variant par exemple de 0 à 6 (0 = "non pertinent" et 6 = "tout à fait pertinent"). Par exemple, les

. – /

avatars *tourisme sportif* et *tourisme culturel* sont respectivement représentés par les vecteurs :

$$\textit{tourisme sportif} = \{gastronomie = 0, snack = 0, patrimoine = 0, art = 0,$$
$$sport\ outdoor = 6, sport\ indoor = 6\};\ \text{et}$$
$$\textit{tourisme culturel} = \{gastronomie = 3, snack = 0, patrimoine = 6, art = 6,$$
$$sport\ outdoor = 0, sport\ indoor = 0\}.$$

et les prestations touristiques "kayak" et "musée d'Annecy" sont respectivement représentées par les vecteurs :

$$\textit{kayak} = \{gastronomie = 0, snack = 0, patrimoine = 0, art = 0,$$
$$sport\ outdoor = 6, sport\ indoor = 0\};\ \text{et}$$
$$\textit{musée} = \{gastronomie = 0, snack = 0, patrimoine = 6, art = 6,$$
$$sport\ outdoor = 0, sport\ indoor = 0\}.$$

La note associée à tout couple $utilisateur - prestation$ est alors définie comme le résultat du produit scalaire entre le vecteur descripteur de l'avatar de l'utilisateur et le vecteur descripteur de la prestation. Par exemple, la note d'un utilisateur $sportif$ sur la prestation $kayak$ est 36 tandis que la note de ce même utilisateur sur la prestation *musée* est de 0 : cet utilisateur préfère le kayak au musée.

4. Formalisation du système de recommandation

Le système de recommandation est formalisé par une matrice $\mathbf{N}$ de notes contenant à la fois les notes explicitement fournies par les utilisateurs et celles générées à partir des avatars de ceux-ci. La figure 1 représente cette matrice avec en vert les notes des utilisateurs et en jaune celles des avatars.

	Objet 1	Objet 2			Objet O	
Utilisateur 1	1	2	3	1	1	5
Utilisateur 2	2	2	3	1	1	5
	3	2	2	1	1	5
	1	3	3	4	5	3
Utilisateur U	1	3	3	5	4	3

Figure 1. Représentation matricielle de l'état du système de recommandation avec en vert les notes explicites des utilisateurs et en jaune celles des avatars.

Dans le reste de ce document, nous distinguons les notes explicites obtenues des utilisateurs de celles des avatars et nous notons N_{ult} l'ensemble des notes des utilisateurs et N_{avt} l'ensemble des notes des avatars.

Le principe de recommandation consiste à suggérer aux utilisateurs les objets ayant les meilleures notes. Cependant, comme les notes des avatars sont intrinsèquement approximatives, cette approche peut mener à des recommandations peu pertinentes. Nous proposons une méthode permettant d'adapter les notes des avatars en utilisant celles explicitement obtenues des utilisateurs. La méthode proposée consiste à grouper les utilisateurs ayant des intérêts communs et à appliquer pour chacun de ces groupes la méthode des facteurs latents pour permettre un rétrocontrôle des notes des utilisateurs sur les notes des avatars. L'opération de partitionnement des utilisateurs permet d'améliorer la précision de la méthode des facteurs latents en minimisant son erreur quadratique moyenne.

5. Adaptation des notes des avatars

Dans cette section, nous décrivons la méthode employée pour adapter les notes des avatars dans l'objectif de personnaliser les avatars de chaque utilisateur et de mieux définir les intérêts des utilisateurs envers les objets.

La méthode proposée, permettant d'adapter les notes des avatars en utilisant celles des utilisateurs, consiste à :

1. partitionner les utilisateurs, les lignes de la matrice $\mathbf{N}$, en groupes homogènes de manière à diviser cette dernière en k sous-matrices $\mathbf{N_k}$ en utilisant la méthode des *k-moyennes* (Arthur, Vassilvitskii, 2007 ; Kodinariya, Makwana, 2013);

2. effectuer une factorisation $\mathbf{P_k}{\mathbf{Q_k}}^{\mathbf{T}}$ de ces sous-matrices (Koren *et al*., 2009); et à

3. mettre à jour les notes des avatars selon l'équation :

$$\forall k,\ \forall (u,o), u \in \{1,\ \ldots,\ card(U)\} o \in \{1,\ \ldots,\ card(O) : r_{u,o} \in N_{avt}$$

$$r_{u,o} \leftarrow \alpha Avt_{u,o} + (1-\alpha) p_{k,u} q_{k,o}^{T}.$$

$p_{k,u}$ et $q_{k,o}$ représentent respectivement les vecteurs des matrices $\mathbf{P_k}$ et $\mathbf{Q_k}$ si $r_{u,o}$ appartient à la partition k; et $Avt_{u,o}$ représente la note générée par l'avatar de l'utilisateur u pour l'objet o en calculant le produit scalaire de leurs vecteurs descripteurs respectifs tels que décrits dans la section 3. Le coefficient $\alpha \in [0;\ 1]$ permet de contrôler l'influence de l'estimation dans le processus d'adaptation et, au cours des mises à jour successives des notes, d'oublier la note initiale de l'avatar.

6. Conclusion

L'approche que nous proposons est assez générique pour être expérimentée dans des domaines d'application différents. L'introduction de la notion d'avatar est intéressante pour permettre d'initialiser le système et apporte une réponse à la question du

démarrage à froid. Notre proposition permet également d'adapter dynamiquement les notes initialement générées par les avatars pour les rendre de plus en plus pertinentes pour les utilisateurs.

Une des limites du cas d'étude telle que réalisée actuellement est que l'identification des descripteurs et l'attribution des poids à ces derniers sont réalisées manuellement. Cette question peut faire l'objet de perspectives à ce travail.

Remerciements

Ces travaux ont été en partie financés par le projet Interreg Franco-Suisse Transfrontour (2016-2019) sur des fonds FEDER.

Bibliographie

Arthur D., Vassilvitskii S. (2007). k-means++: The advantages of careful seeding. In *Proc. of the 18th annual acm-siam symposium on discrete algorithms*, p. 1027 - 1035.

Béchet N. (2016). *état de l'art sur les systèmes de recommandation*. Rapport technique. INRIA.

Benouaret I. (2017). *Un système de recommandation contextuel et composite pour la visite personnalisée de sites culturels*. Thèse de doctorat non publiée, Université de Technologie de Compiègne.

Chen G., Chen L. (2015, août). Augmenting service recommender systems by incorporating contextual opinions from user reviews. *User Modeling and User-Adapted Interaction*, vol. 25, n° 3, p. 295–329. Consulté sur https://doi.org/10.1007/s11257-015-9157-3

Hernández-Rubio M., Cantador I., Bellogín A. (2018, novembre). A comparative analysis of recommender systems based on item aspect opinions extracted from user reviews. *User Modeling and User-Adapted Interaction*. Consulté sur https://doi.org/10.1007/s11257-018-9214-9

Kodinariya T. M., Makwana P. R. (2013). Review on determining number of cluster in k-means clustering. *International Journal of Advance Research in Computer Science and Management Studies*, vol. 1, n° 6, p. 90 - 95.

Koren Y., Bell R., Volinsky C. (2009). Matrix factorization techniques for recommender systems. *IEEE Computer Society*, vol. 9, n° 1, p. 42 - 49.

Negre E., Ravat F., Teste O., Tournier R. (2013). Problème du démarrage à froid pour les système de recommandation. In *Proc. of inforsid 2013*, p. 439 - 454.

Pazzani M. J., Billsus. (2007). The adaptive web: Methods and strategies of web personalization. In P. Brusilovsky, A. Kobsa, W. Nejdl (Eds.),, vol. 4321, p. 325-341. Springer-Verlag, Berlin Heidelberg New York.

Schafer J. B., Frankowski D., Herlocker J., Sen S. (2007). The adaptive web: Methods and strategies of web personalization. In P. Brusilovsky, A. Kobsa, W. Nejdl (Eds.),, vol. 4321, p. 291-324. Springer-Verlag, Berlin Heidelberg New York.

Un cadre d'aide à l'exploitation des résultats de prédictions, à destination d'experts de domaine

Gabriel Ferrettini, Julien Aligon, Chantal Soulé-Dupuy

Université de Toulouse, UT1, IRIT (CNRS/UMR 5505), Toulouse, France
prenom.nom@irit.fr

RÉSUMÉ. *L'apprentissage automatique (ML) s'est révélé de plus en plus essentiel dans de nombreux domaines. Pourtant, de nombreux obstacles limitent encore son utilisation par des non-experts. Au premier rang de ceux ci se situe le manque de confiance dans les résultats obtenus et a inspiré plusieurs approches explicatives dans la littérature. Nous proposons ici un cadre pour exploiter cette capacité à expliquer les prédictions de ML de manière simple. Ceci a pour but de permettre aux outils ML existants de fournir une information plus interprétable aux utilisateurs ne maîtrisant pas encore l'apprentissage automatique. Ceci est effectué en fournissant à l'utilisateur une explication détaillée de l'influence des attributs pour chaque instance prédite, en relation avec le modèle d'apprentissage automatique. Nous montrerons également en quoi cette explication aide les utilisateurs non-experts à effectuer certaines tâches d'analyse complexes, telles que la sélection de modèles et l'ingénierie de fonctionnalités, et fournit une assistance pour exploiter efficacement les résultats d'un modèle prédictif.*

ABSTRACT. *Machine learning (ML) has proven increasingly essential in many fields. Yet, a lot obstacles still hinder its use by non-experts. The lack of trust in the results obtained is foremost among them, and has inspired several explanatory approaches in the litterature. We propose here a framework to build upon this ability to explain ML predictions in a simple way. This aims to allow already existing ML tools to provide a more interpretable information to users not already knowledgeable in machine learning. This is performed by providing the user a detailed explanation of the attributes influence for each single predicted instance, related to the machine learning model . We will also show how this explanation helps non-expert users to perform some complex analysis tasks, such as model selection and feature engineering, and provides assistance in exploiting efficiently the results of a predictive model.*

MOTS-CLÉS : *apprentissage automatique, explication de prédictions*

KEYWORDS: *machine learning, prediction explanation*

1. Introduction

Dans la plupart des cas, l'analyse de données suppose des compétences très spécifiques dans la réalisation et l'utilisation de modèles. Par exemple, concevoir un modèle prédictif (modèle finalement très commun et populaire) requiert une certaine expertise dans le domaine de la fouille de données. Ainsi, ces tâches d'analyse sont particulièrement difficiles à appréhender pour des utilisateurs experts de domaine (i.e. ayant une connaissance profonde des données à analyser, sans pour autant avoir des compétences en fouille de données). Un autre problème, pour les utilisateurs non experts, porte sur l'exploitation des résultats fournis par les analyses. En effet, ces résultats n'offrent généralement aucune indication sur la manière dont ils ont été produits, ce qui peut fortement impacter la confiance de ces utilisateurs sur la pertinence des analyses. Ainsi, dans le but de faire confiance aux résultats produits (par des méthodes qu'il ne connaît pas nécessairement) et les appliquer, un utilisateur expert du domaine aura tendance à les rattacher, d'une manière ou d'une autre, à sa propre connaissance de ce domaine.

L'ambition de nos recherches est d'aider les utilisateurs experts de domaine à (re)trouver une motivation pour s'impliquer dans des opérations d'analyse de données en donnant un sens aux processus utilisés. Notamment, notre objectif est de s'appuyer autant que possible sur leurs domaines d'expertise, tout en limitant les connaissance en analyse de données. Plus précisément, nous concentrons nos travaux sur des tâches de classification. Le but est de permettre à un utilisateur final d'effectuer des opérations de sélection de modèle prédictif et de spécifier les caractéristiques du modèle à l'aide des connaissances de domaine (nommée $feature\ engineering$) qui sont des tâches d'analyse connues pour être complexes. L'idée principale est que l'utilisateur n'ait qu'à exercer son expertise critique sur les classifications proposées. Afin de comprendre "comment" ou "pourquoi" une prédiction particulière est faite, nous proposons à l'utilisateur plusieurs types d'explications du comportement du modèle sur ces instances. Notre objectif final est également d'aider l'utilisateur à exploiter le modèle prédictif qu'il a produit. A cette fin, nous proposons donc une méthode de type $sandbox$ pour l'exploitation de modèle, permettant d'étudier le comportement de celui-ci sur de nouvelles instances simulées.

Les contributions, présentées dans ce papier, sont composées de:

– Un cadre d'aide à l'exploitation des résultats de prédictions à destination d'experts de domaine, en s'appuyant sur les travaux de (Strumbelj, Kononenko, 2010) et (Ribeiro *et al*., 2016), .

– Une proposition pour une nouvelle méthode d'explication de classification, basée sur (Strumbelj, Kononenko, 2010).

Ce papier est organisé de la manière suivante. La section 2 présente notre positionnement par rapport à l'état de l'art, portant sur l'aide à l'analyse de données et les explications de prédictions. Par la suite, les sections 3 et 4 développent notre cadre général pour l'aide à l'exploitation de prédictions, en introduisant les concepts de *Dataset*, *Workflow* et *Explication de prédictions*, ainsi qu'un exemple illustrant notre

démarche. Enfin, la section 5 discute des futures expériences à réaliser, permettant de vérifier nos hypothèses énoncées précédemment, et la section 6 conclut ce papier par plusieurs perspectives de travail.

2. Positionnement

Les systèmes existants et les outils pour l'apprentissage automatique et l'analyse de données se concentrent essentiellement sur la mise à disposition et l'explication des *méthodes* et des $algorithmes$. Cette approche est particulièrement utile pour des utilisateurs experts en fouille, mais requiert encore une connaissance avancée en analyse de données. En effet, la plupart des plateformes bien connues en analyse de données, comme Weka (Hall *et al*., 2009) ou Knime (Berthold *et al*., 2007) fournissent des descriptions détaillées des méthodes et algorithmes qu'ils comportent, illustrés souvent d'exemples. Malheureusement, ces descriptions détaillées sont un substitut très limité à la compréhension des informations en analyses de données.

Quelques plateformes, comme par exemple RapidMiner (Hofmann, Klinkenberg, 2013), Orange (Demšar *et al*., 2013), ont tout de même consacré une attention particulière à la présentation et à l'explication de *résultats* d'analyse à l'utilisateur. En fournissant des interfaces de visualisation bien conçues, ces platesformes aident l'utilisateur à comprendre les résultats obtenus, ce qui constitue un premier pas vers une utilisation compréhensible des modèles. Toutefois, ces outils ne peuvent toujours pas expliquer comment les résultats de fouille ont été obtenus, ce qui reste un facteur dissuasif et important pour des utilisateurs de domaine, où de mauvaises décisions peuvent avoir des conséquences graves. Aider ces utilisateurs à comprendre $pourquoi$ une prédiction particulière est faite, de sorte à pouvoir vérifier son raisonnement, pourrait renforcer leur confiance dans un modèle, ou au contraire leur donner une raison valable de le rejeter. Cette intution montre bien la nécessité d'expliquer les prédictions.

Plus récemment, l'outil de Google *What if*[1], principalement basé sur le travail de (Wachter *et al*., 2017), propose de nombreux outils d'exploration pour l'apprentissage automatique. Ceux-ci aident un utilisateur à comprendre et à exploiter les modèles de manière intuitive. Cela se fait principalement en permettant à l'utilisateur d'explorer l'espace de prédictions, à partir d'un modèle entrainé, et en affichant différentes métriques d'une manière facilement interprétable.

Notre travail vise justement à prendre en charge de telles fonctionnalités avec de nouvelles informations plus proches des connaissances du domaine de l'utilisateur : nous fournissons à l'utilisateur une mesure simple et interprétable de l'influence de chaque attribut (d'un ensemble de données) sur une prédiction. Grâce à cette méthode, nous souhaitons l'aider à comprendre le modèle d'apprentissage automatique, non pas de manière globale (existant dans beaucoup d'outils), mais par l'explication des

1. https://pair-code.github.io/what-if-tool/

éléments composant les prédictions (les instances). Cette méthode permettra de mieux appréhender un modèle donné.

L'une des contributions les plus significatives s'orientant dans cette voie peut être trouvé dans (Strumbelj, Kononenko, 2010). Dans ce papier, les auteurs décident de prendre en compte les interactions entre chaque attribut d'un ensemble de données afin d'approcher d'avantage leur influence réelle. Cet article mène par la suite au travail de (Lundberg, Lee, 2017), lequel a théorisé les méthodes d'explications dites "additives", dont la méthode proposée par (Strumbelj, Kononenko, 2010) fait partie. Cet article a notamment mis en lumière plusieures propriétés intéressantes de ces méthodes, qui en font un objet théorique très utile.

Les propriétés d'explication de prédictions ont été principalement étudiées par (Ribeiro *et al*., 2016). D'après ce papier, l'intérêt d'expliquer des modèles à l'utilisateur est triple :

– Premièrement, cela peut être perçu comme un moyen de comprendre le fonctionnement d'un modèle en observant comment il se comporte à divers endroits de l'espace des instances.

– Deuxièmement, cela peut aider un utilisateur non expert à juger de la qualité d'une prédiction et même à identifier la cause des problèmes dans sa classification. Les corriger conduirait alors l'utilisateur à effectuer certaines opérations de *feature engineering*.

– Troisièmement, cela permet à l'utilisateur de décider du modèle lui semblant préférable à un autre, même s'il n'a pas toujours connaissance des principes sous-jacents de chacun d'eux. Il suffit d'expliquer à l'utilisateur le comportement de mêmes instances, classées selon différents modèles.

La capacité d'expliquer une prédiction de n'importe quel modèle semble donc être un élément capital pour permettre à un public plus large, non expert, d'accéder aux modèles d'apprentissage automatique et de les utiliser. Ce besoin nous a amené à considérer les différents systèmes d'explication développés dans la littérature comme ayant un intérêt majeur pour la recherche d'une certaine autonomie des experts de domaine effectuant des activités d'analyse de données. Ainsi, le point clé de ce travail est que chaque choix significatif, effectué par l'utilisateur au cours d'une tâche de fouille de données, doit s'appuyer le plus possible sur la connaissance approfondie qu'il possède de son propre domaine. La section suivante donne un exemple concret de la manière dont cet objectif peut être atteint.

3. Définitions préliminaires et Méthodes d'explication

Dans cette section, nous définissons les éléments important de notre cadre que sont le Dataset, Workflow et l'Explication de Prédictions.

3.1. Dataset et Workflow

Un Dataset est défini comme un ensemble d'instances décrites selon des attributs. Soit $A = \{a_1, ..., a_n\}$, les attributs d'un dataset, une instance x est un vecteur de n valeurs d'attributs de A.

Un Workflow, dans la forme la plus générale, consiste en un graphe orienté d'opérations d'analyse de données (Serban *et al*., 2013). Il peut inclure la plupart des étapes d'analyse de données, tels que le prétraitrement (nettoyage des données, normalisatin, etc.), la construction de modèles, la recherche de motifs, et même l'optimisation de paramètres pour d'autres opérations de fouille. Concernant les méthodes d'explication, nous considérons, dans ce papier, seulement celles applicables aux classifications supervisées. Ainsi nous ne considèrerons que des workflows résultant de ces types de modèles.

3.2. Explication de Prédictions

Soit un dataset D et un ensemble de n attributs $A = \{a_1, ..., a_n\}, \forall i \in [1..n], a_i \subset \mathbb{R}$. Chaque instance $x \in D$ est définie par les valeurs de chacun de ses attributs : $x = \{x_1, ..., x_n\}, \forall i \in [1..n], x_i \in a_i$. Nous voulons expliquer un modèle prédictif, basé sur la fonction $f : A \rightarrow [0, 1]$, dont le résultat est un score de confiance d'une classification d'une instance x pour la classe C, prédite par le modèle.

Une des premières définitions de l'explication de classification a été proposée dans (Štrumbelj, Kononenko, 2008). Selon leur méthode, l'influence d'un attribut a_i sur la classification d'une instance donnée peut être définie comme la différence entre la prédiction du classifieur (avec a_i) et sa prédiction sans la connaissance de l'attribut a_i. Ainsi, étant donné un ensemble d'instances décrit selon des attributs de A, l'influence de l'attribut a_i sur la classification d'une instance x par la fonction de confiance du classifieur f sur la classe C peut être représentée comme:

$$inf^C_{f,a_i}(x) = f(x) - f(x \backslash a_i) \tag{1}$$

Où $f(x \backslash x_i)$ représente la probabilité, selon le modèle étudié que l'instance x fasse partie de la classe c sans connaître l'attribut a_i.

Afin de simuler cette absence d'attribut a_i, les auteurs de (Štrumbelj, Kononenko, 2008) théorisent trois approches possibles parmi lesquelles nous avons sélectionné les deux plus génériques pour nos tests:

– Réentrainement du classifieur sur le dataset après suppression de l'attribut a_i.

– Utilisation d'une solution approchante à l'aide d'une espérance mathématique selon l'équation 2, comme proposé dans (Robnik-Šikonja, Kononenko, 2008).

$$f(x \backslash a_i) = \sum_{y \in val(a_i)} p(a_i = y) f(x \leftarrow a_i = y) \tag{2}$$

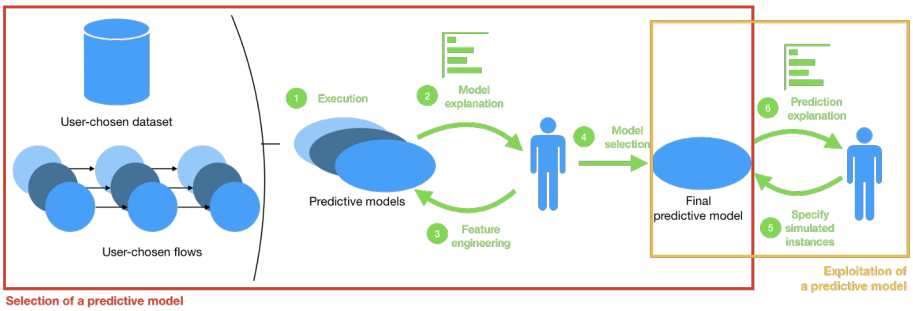


FIGURE *1. Construction et utilisation d'un modèle prédictif*

L'équation 2 consiste en une somme des distributions de probabilités du classifieur f pour une instance x, avec l'attribut a_i prenant toutes ses valeurs possibles, pondérées par la probabilité que l'attribut prenne cette valeur dans le jeu de données d'origine. Ces explications ont de nombreuses applications possibles pour aider les utilisateurs non experts à comprendre et à travailler dans un cadre d'apprentissage automatique. Nous allons démontrer un certain nombre de ces applications dans l'exemple suivant.

4. Exemple illustrant notre démarche

Notre exemple est séparé en deux cas d'utilisation. Dans la section 4.1, nous montrons comment un utilisateur expert de domaine peut être guidé tout au long du processus complexe de construction d'un modèle prédictif, tandis que la section 4.2 illustre comment des explications peuvent apporter de nouvelles connaissances lors de l'exploitation d'un modèle prédictif. Ces deux processus sont représentés dans la Figure 1 et sont basés sur les fonctionnalités les plus courantes de la litterature décrites Section 2.

4.1. Construction d'un nouveau modèle

Le Docteur Smith, un scientifique travaillant dans le domaine de la botanique, a rassemblé un dataset conséquent sur différentes fleurs qu'il a classées par espèce. Son objectif est de comprendre les principales caractéristiques de ces espèces afin de les reconnaître plus facilement par la suite et donc de faciliter son travail dans le futur. Le cadre que nous proposons devrait pouvoir l'aider dans sa tâche de classification.

1. Exécution - Le système propose un ensemble de workflows disponibles et capables de produire des modèles prédictifs. Smith ne connaissant pas grand-chose en apprentissage automatique, il fournit son jeu de données et sélectionne cinq workflows différents à l'aide de leurs descriptions respectives. Ces dernières le renseignent sur les traitements employés pour construire le modèle (e.g. prétraitement des données),

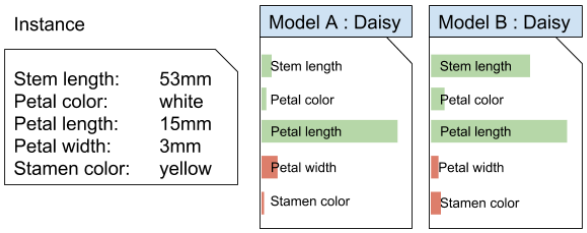


FIGURE 2. *Explications de prédictions*

ainsi que le type même de modèle (e.g. arbre de décision, Bayes naïf) sous forme textuelle. Le système exécute ensuite les cinq workflows sur le jeu de données de Smith, générant cinq modèles prédictifs différents.

2. Explication de modèle - À l'aide de ces modèles, le système peut générer la classification d'une instance donnée du dataset et fournir l'explication afférente à chaque modèle. Ces explications prennent la forme d'influences d'attributs. Par exemple, dans la figure 2, on informe le Docteur Smith qu'une fleur particulière est classée comme espèce S par les modèles A et B, mais A a pris cette décision compte tenu principalement de la longueur des pétales de la fleur, tandis que B a également pris en compte la longueur de la tige. Smith, considérant que la longueur de la tige de cette fleur particulière n'est pas réellement caractéristique de son espèce, il sera donc plus enclin à faire confiance au modèle A. Une manière d'expliquer simplement un modèle est de fournir un ensemble d'instances (les plus diverses possibles) afin de donner un bon aperçu du comportement de chaque modèle. Un algorithme est décrit dans ce sens par (Ribeiro *et al.*, 2016).

3. Feature engineering - A l'aide de ces explications, Smith se rend compte que l'utilisation d'un attribut, $ID\ Flower$, dans son dataset était une erreur : chaque modèle le considère comme très important pour sa prédiction (chaque modèle apprend donc que la fleur avec un identifiant idX appartient à une espèce y). Il est fort probable que l'identifiant de la plupart des fleurs est corrélé à son espèce dans le dataset. A l'aide de l'outil, il peut sélectionner cet attribut "ID Flower" et demander au système de le supprimer. Le système passe ensuite à l'étape (2) une seconde fois. Smith peut ensuite examiner les nouvelles influences générées et décider s'il souhaite exclure un autre attribut.

4. Sélection de modèle - Satisfait de ses modifications, Smith fait ensuite attention aux différences de raisonnement entre les modèles et se rend compte que, si l'un était certes plus précis dans ses classifications, un autre était plus "logique" dans l'utilisation de chaque attribut. En effet, ce dernier modèle donne plus d'importance à la longueur de la tige de fleur ($stem\ lenght$), quand cela est pertinent, et met de côté plus facilement les attributs dont Smith sait qu'ils ne sont pas si importants. Cette vérification de la pertinence de chaque modèle, à l'aide des connaissances du domaine, montre comment Smith peut décider du modèle à utiliser dans la pratique tout en se

fiant uniquement à ses propres connaissances. Le système génère ensuite le workflow correspondant au modèle choisi sur le dataset et le stocke pour une utilisation ultérieure. Ce workflow est choisi parmi ceux déjà réalisés par le passé. Un système de recommandation par filtrage collaboratif est alors nécessaire, comme par exemple, celui proposé dans (Raynaut, 2018).

4.2. Explorer et exploiter un modèle entrainé

Ce deuxième cas d'utilisation reprend l'exemple du docteur Smith après quelques mois. Après avoir étudié et répertorié les fleurs dans une nouvelle région du monde, il souhaite comparer sa population de fleurs à celle qu'il a étudiée dans le premier cas d'utilisation. Le modèle prédictif qu'il a produit à ce moment-là peut maintenant être très utile, lui confiant ainsi la tâche coûteuse d'identifier l'espèce de chaque fleur. Bien qu'il s'agisse de l'utilisation la plus courante d'un modèle prédictif, notre objectif ici est de montrer comment notre cadre permet d'aller encore plus loin. En effet, le but de ce cas d'utilisation est de permettre d'explorer et de présenter les caractéristiques apprises par le modèle pendant son entrainement, afin de trouver de nouveaux liens et corrélations possibles, non identifiés auparavant.

1. Après avoir collecté sur le terrain des fleurs dont les longueur et largeur de pétales sont inhabituels, le docteur Smith souhaite étudier l'importance que le modèle accordera à ces combinaisons d'attributs. L'idée est de pouvoir créer une instance "simulée", donc non présente initialement dans le dataset, afin de tester le comportement du modèle. Smith peut alors affecter aux attributs de longueur et de largeur de pétale une de ces combinaisons inhabituelles qu'il a collectées et demander au système de calculer la prédiction et l'explication.

2. Le système classe l'instance simulée avec le modèle choisi lors du premier cas d'utilisation et génère l'explication de cette classification en indiquant l'influence de chaque attribut que le Dr. Smith souhaite étudier. Avec cette nouvelle information, il peut itérativement concevoir une nouvelle instance simulée, en repartant de l'étape (5), et étudier l'influence des attributs vis à vis de ces nouvelles valeurs. Cette phase peut ainsi aider à l'identification de nouveaux points saillants pour ses recherches.

Le but principal de notre cadre est là : aider à extraire de manière interactive de nouvelles informations, à partir de modèles prédictifs standards. Notre cadre peut ainsi être vu comme un catalyseur pour l'analyse de différents types de données.

5. Comparaison des différentes méthodes d'explication

5.1. Premières expérimentations

5.1.1. Raisonnement préliminaire

Notre but est de comparer et évaluer les deux méthodes d'explication définies Section 3.2. En particulier, l'approche statistique a été préférée dans la littérature (eg.

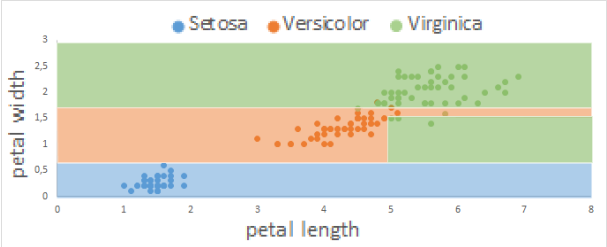


FIGURE *3. Répartition des trois différentes classes par longueurs et largeurs de pétales, avec leur généralisation correspondante à l'arbre de décision*

(Štrumbelj, Kononenko, 2008) and (Strumbelj, Kononenko, 2010)), en raison du coût de calcul potentiellement très élevé de la méthode par réentrainement $(\bigcirc(l * n))$, avec n le nombre d'attributs et l la complexité du modèle d'apprentissage).

5.1.2. Première expérience : exécution

Pour une expérience préliminaire, et afin de faciliter les interprétations, nous appliquons les deux méthodes à un simple arbre de décision ID3, (Quinlan, 1986), formé à partir du dataset *Iris* de Fisher [2]. Comme un arbre de décision est un modèle naturellement interprétable et que le jeu de données *Iris* est bien étudié dans la littérature, il est facile de comparer et d'interpréter les explications générées à partir de modèles prédictifs et de détecter les problèmes éventuels causés par les deux méthodes.

Nous utilisons Weka (Hall *et al.*, 2009) et OpenML (Vanschoren *et al.*, 2014) pour la gestion des données et la création des modèles tout en garantissant la reproductibilité de toutes les expériences. Afin d'estimer la fiabilité des deux modèles, nous appliquons une validation croisée (*5-fold cross validation*) sur l'ensemble des données. Les explications sont générées sur l'ensemble de validation à chaque itération de la validation croisée, pour la méthode statistique et par réentrainement. Nous générons ainsi les explications des prédictions du classifieur d'arbres *J48* de Weka pour l'ensemble des données *Iris*. Nous pouvons ensuite comparer ces explications à l'arbre de décision et obtenir une idée générale de l'exactitude des explications.

5.1.3. Première expérience : résultat et interprétation

Chaque instance du dataset *Iris* est composé de quatre attributs: longueur de pétale (*petal length*), largeur de pétale (*petal width*), longueur de sépale (*sepal length*) et largeur de sépale (*sepal width*). Chacune des instances peut faire partie de l'une de ces trois classes : *Iris Setosa*, *Versicolor* ou *Virginica*.

2. Iris, Fisher : https://en.wikipedia.org/wiki/Iris_flower_data_set

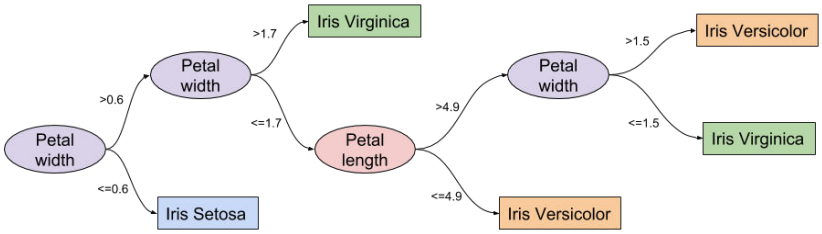


FIGURE *4. Un arbre de décision entrainé sur Iris*

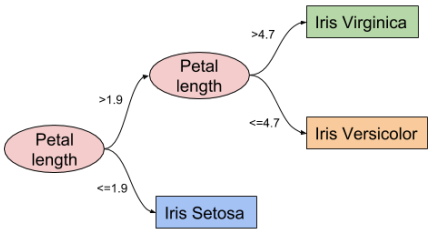


FIGURE *5. Identique à Figure 4 sans l'attribut largeur de pétale (petal width)*

Un simple coup d'œil à l'arbre de décision présenté dans la Figure 4 indique que l'influence doit être nulle pour les attributs longueur et largeur des sépales, car ils ne sont pas du tout utilisés par l'arbre. De plus, les instances de la classe *Setosa* ne devraient être influencées que par la largeur des pétales, car c'est le seul attribut utilisé pour les classer. Pour *Iris Virginica* et *Versicolor*, on peut s'attendre à une influence importante de la largeur des pétales et à une influence plus faible, mais néanmoins significative, de la longueur des pétales, car ces deux attributs sont utilisés (la largeur des pétales demeurant prédominante dans la classification des instances). Nous pouvons maintenant comparer ces suppositions aux résultats du tableau 6.

	Training method				Statistical method			
	Sepal length	Sepal width	Petal length	Petal width	Sepal length	Sepal width	Petal length	Petal width
Setosa	0	0	0	0	0	0	0.005	0.995
Versicolor	0	0	0.785	0.135	0	0	0.269	0.731
Virginica	0	0	-0.0283	0.615	0	0	0.171	0.824
Average	0	0	0.253	0.256	0	0	0.148	0.850

FIGURE *6. Influence moyenne des différents attributs, selon chaque méthode d'explication, pour chaque classe d'instance*

Nous pouvons facilement voir que la méthode par réentrainement ne se comporte pas comme prévu pour les instances de *Setosa*, car tous les attributs reçoivent une influence égale à 0. Cela peut être compréhensible en regardant l'arbre réentrainé sans l'attribut de largeur de pétale dans la figure 5, et la représentation du concept appris par l'arbre de la figure 3. Nous pouvons voir qu'en supprimant l'un des attributs parmi longueur des pétales et largeur des pétales, il reste possible de séparer linéairement la classe *Setosa* des autres en n'utilisant que la longueur de pétale, tout en maintenant une confiance de 100% dans la classification. Ainsi, chaque attribut est considéré comme sans importance par la méthode d'apprentissage lors de la classification des instances de *Setosa*. Cela implique que pour chaque dataset dans lequel deux attributs portent des informations très similaires, la méthode par réentrainement ne pourra pas générer d'explication satisfaisante. Afin de corriger cette aberration et de générer une explication plus précise, nous devons prendre en compte plus que la simple utilité des attributs pris individuellement.

Compte tenu des résultats pour les autres classes, nous ne pouvons pas encore conclure sur les différences entre les influences générées par les deux méthodes, mais nous pouvons voir que le comportement de la méthode statistique semble être plus proche de celui attendu. La simplicité du dataset $Iris$ avantage sans doute la méthode statistique car l'approximation de la répartition d'un petit ensemble d'attributs sera bien entendu plus proche de la situation réelle.

Tout du moins, et au vu de ces résultats préliminaires, il semble donc qu'il y ait une nécessité de considérer non pas les attributs comme des entités indépendantes, mais comme au moins partiellement interdépendantes.

5.2. Intuitions pour de futures expérimentations

5.2.1. Construction de nouvelles méthodes d'explication

Dans le but de répondre aux problèmes d'interaction entre attributs, discuté dans la section précédente, nous proposons de nous inspirer du travail de (Strumbelj, Kononenko, 2010). Nous sommes ici dans un cadre qui se rapproche de la situation d'un jeu dit "à coalitions", où chaque groupes d'attributs peux avoir une influence sur la prédiction du modèle. Nous ne pouvons donc considérer uniquement les attributs seuls mais bien toutes les coalitions possibles d'attributs. L'influence d'un attribut devra se mesurer en fonction de son importance dans chaque coalition. Nous pouvons alors nous rapporter aux jeux de coalition tels que définis par Shapley dans (SHAPLEY, 1953) : Un jeu de coalition de N joueurs est défini comme une fonction de mapping de sous-ensembles de joueurs de gains $g : 2^N \mapsto \mathbb{R}$. Le parallèle peut facilement être établi avec notre situation, où nous souhaitons évaluer l'influence d'un attribut donné *dans toutes les coalitions possibles d'attributs*. Nous examinerions alors non seulement l'influence de l'attribut, mais également son utilisation dans tous les sous ensembles d'attributs. Nous définissons donc l'*influence complète* d'un attribut $a_i \in A$ sur la classification d'une instance x (les notations restent les mêmes que dans 3.2):

$$\mathcal{I}_{a_i}^{C}(x) = \sum_{A' \subseteq A \setminus a_i} shap(A') * (inf_{f,(A' \cup a_i)}^{C}(x) - inf_{f,A'}^{C}(x))) \quad (3)$$

Avec $p(A')$ une fonction de pénalité dépendant de la taille du sous-ensemble A'. En effet, si un attribut change beaucoup le résultat d'un classifieur qui dépend déjà de beaucoup d'attributs, il peut être considéré comme très influent par rapport aux autres. À l'inverse, un attribut qui modifie le résultat d'un classifieur alors que ce classifieur se base sur un petit nombre d'attributs, ne peut pas être considéré comme ayant une influence déterminante.

A cette fin, la valeur de Shapley (SHAPLEY, 1953) est un candidat prometteur, et définis la pénalité comme :

$$shap(A') = \frac{|A'|! * (|A| - |A'| - 1)!}{|A|!} \quad (4)$$

Cette *influence complète* d'un attribut prend désormais en compte son importance parmi toutes les configurations d'attributs possibles, ce qui est plus proche de l'intuition d'origine derrière l'influence des attributs. Cependant, calculer l'*influence complète* d'une seule instance est extrêmement coûteux, avec une complexité de $\bigcirc(2^n * l)$, avec n le nombre d'attributs et l la complexité du modèle à expliquer.

Il n'est donc pas pratique d'utiliser l'*influence complete*. Par conséquent, il devient nécessaire de rechercher un moyen plus efficace d'expliquer les prédictions. Bien que l'*influence complete* soit trop lourde en calculs, son intérêt en fait une excellente baseline de départ, comme le propose d'ailleurs (Strumbelj, Kononenko, 2010). Nous pouvons donc évaluer d'autres méthodes d'explication en étudiant leurs différences par rapport à l'*influence complète*.

5.2.2. Travaux en cours: Trouver de bons estimateurs de l'influence complète

Une approximation de l'*influence complète* pourrait fournir une méthode d'explication à la fois précise et pratique. Certaines ont été proposées dans (Strumbelj, Kononenko, 2010) et (Lundberg, Lee, 2017), bien que ces heuristiques reposent sur plusieurs aprioris : modèle de prédiction linéaire, indépendance des attributs. Cela ne peut être pris en compte dans notre cadre, car nous souhaitons travailler avec n'importe quel ensemble de données et processus. Cela nous conduit donc à rechercher de nouvelles heuristiques.

En particulier, l'évaluation d'un sous-ensemble de tous les sous-groupes d'attributs autorise des limites beaucoup plus pratiques en termes de complexité, tout en produisant des estimateurs à priori plus précis que la simple considération des attributs seuls (*influence linéaire*). Nous pouvons alors considérer une *influence k-complète* (*influence complète* de profondeur k) définie comme :

$$\mathcal{I}_{a_i}^{C_k}(x) = \sum_{A' \subseteq A \setminus a_i | A'| \geq |A| - k} p(A') * (inf(x_{A' \cup \{a_i\}}) - inf(x_{A'})) \quad (5)$$

D'ailleurs, nous pouvons noter que l'*influence linéaire* est alors identique à l'influence de l'*influence* 1-*complète*.

Une autre approche possible consiste à identifier les attributs ayant une relation entre eux, en utilisant un algorithme de regroupement d'attributs (comme dans (Henelius *et al*., 2014)).

Nous pouvons obtenir un groupe tel que $G = \{\{a_1, a_3\}, \{a_2, a_5, a_8\}, \{a_4\}...\}$. Avec de tels regroupements d'attributs, il devient possible de ne considérer que les relations entre les attributs d'un sous-groupe, sans avoir à prendre en compte toutes les combinaisons d'attributs possibles.

Nous obtenons alors une *influence coalitionnelle* d'un attribut $a_i \in g, g \in G$:

$$simple\mathcal{I}^C_{a_i}(x) = \sum_{g' \subseteq g \setminus a_i} shap(g') * (inf^C_{f,(g' \cup a_i)}(x) - inf^C_{f,g'}(x))) \qquad (6)$$

Étant donné que nous pouvons définir un cardinal maximum c pour nos sous-groupes, la complexité serait désormais, dans le pire des cas, $O(2^c * \frac{n}{c} * l) \approx O(n * l)\ avec\ l = m(n)$

Afin de déterminer s'il est possible de générer une approximation satisfaisante de l'influence d'un attribut avec la nouvelle *influence* k-*complète* et l'*influence coalitionnelle*, il faut évaluer le nombre des combinaisons d'attributs à prendre en compte avant d'être suffisamment proches de l'*influence complète* définie dans 3. De plus, nous devons évaluer si le résultat de l'*influence* k-*complète* produit bien de meilleures explications que l'*influence linéaire* et l'*influence coalitionnelle*, au vu de son coût de calcul plus élévé.

Pour nos expériences, nous devrons rassembler un ensemble de datasets comportant relativement peu d'attributs, afin de maintenir des limites réalisables en termes de complexité. Pour chaque dataset, l'*influence complète* sera calculée, ainsi que chaque approximation par l'*influence* k-*complète* et *influence coalitionnelle*. Enfin, l'influence calculée par l'*influence linéaire* sera également générée. Chaque méthode sera ensuite comparée à la méthode complète.

Ces résultats nous diront tout d'abord s'il y a un réel avantage à utiliser l'une des méthodes alternatives et dans quels cas une méthode est préférable aux autres.

6. Conclusion

Nous avons proposé dans cet article un cadre d'explication des modèles prédictifs visant à présenter l'utilisateur expert de domaine comme un élément actif du processus de construction de modèle.

Les différentes expériences que nous proposons sont une étape supplémentaire vers l'aide à l'analyse de données, en particulier l'explication de modèles en appren-

tissage automatique. Notre première tentative de comparaison des méthodes de réentrainement et des méthodes statistiques nous a conduit à identifier des défauts dans leurs conceptions intrinsèques. En prenant en compte les combinaisons d'attributs, nous pensons proposer une base de référence adéquate pour l'explication de prédictions et espérons obtenir de meilleures explications en termes de précision et de réduction des coûts. L'expérience en cours servira à comparer les différentes méthodes et ses résultats nous aideront à concentrer nos efforts sur les candidats les plus prometteurs. La prochaine étape serait ensuite de développer un outil complet sur cette baseline, dans le but de guider un utilisateur non expert tout au long de la construction et de l'exploitation d'un modèle d'apprentissage automatique.

Une perspective à plus long terme porte aussi sur la problématique d'évaluation des différentes méthodes d'explication de la littérarture. En effet, à notre connaissance, il n'existe pas de benchmark permettant d'évaluer objectivement différentes méthodes d'explication. Ce benchmark pourrait inclure, en plus des datasets, un ensemble d'indicateurs liés à la perception qu'en a l'expert de domaine (Kappa de Cohen ou Kappa de Fleiss par exemple, (Cohen, 1960), (Fleiss *et al*., 1971))

Bibliographie

Berthold M. R., Cebron N., Dill F., Gabriel T. R., Kötter T., Meinl T. *et al*. (2007). KNIME: The Konstanz Information Miner. In *Studies in classification, data analysis, and knowledge organization (gfkl 2007).* Springer.

Cohen J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, vol. 20, n° 1, p. 37-46. Consulté sur https://doi.org/10.1177/001316446002000104

Demšar J., Curk T., Erjavec A., Gorup Črt, Hočevar T., Milutinovič M. *et al*. (2013). Orange: Data mining toolbox in python. *Journal of Machine Learning Research*, vol. 14, p. 2349-2353. Consulté sur http://jmlr.org/papers/v14/demsar13a.html

Fleiss J. *et al*. (1971). Measuring nominal scale agreement among many raters. *Psychological Bulletin*, vol. 76, n° 5, p. 378–382.

Hall M., Frank E., Holmes G., Pfahringer B., Reutemann P., Witten I. H. (2009). The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, vol. 11, n° 1, p. 10–18.

Henelius A., Puolamaki K., Boström H., Asker L., Papapetrou P. (2014). A peek into the black box : exploring classifiers by randomization. *Data mining and knowledge discovery*, vol. 28, n° 5-6, p. 1503–1529. (QC 20180119)

Hofmann M., Klinkenberg R. (2013). *Rapidminer: Data mining use cases and business analytics applications*. Chapman & Hall/CRC.

Lundberg S., Lee S.-I. (2017). A unified approach to interpreting model predictions. In *Nips.*

Quinlan J. (1986, 01 Mar). Induction of decision trees. *Machine Learning*, vol. 1, n° 1, p. 81–106. Consulté sur https://doi.org/10.1023/A:1022643204877

Raynaut W. (2018). *Perspectives de Méta-Analyse pour un Environnement d'aide à la Simulation et Prédiction*. Thèse de doctorat, Université de Toulouse, Toulouse, France. Consulté sur ftp://ftp.irit.fr/IRIT/SIG/2018_These_Raynaut.pdf

Ribeiro M. T., Singh S., Guestrin C. (2016). "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, p. 1135–1144. New York, NY, USA, ACM. Consulté sur http://doi.acm.org/10.1145/2939672.2939778

Robnik-Šikonja M., Kononenko I. (2008, mai). Explaining classifications for individual instances. *IEEE Trans. on Knowl. and Data Eng.*, vol. 20, n° 5, p. 589–600. Consulté sur http://dx.doi.org/10.1109/TKDE.2007.190734

Serban F., Vanschoren J., Kietz J.-U., Bernstein A. (2013, juillet). A survey of intelligent assistants for data analysis. *ACM Comput. Surv.*, vol. 45, n° 3, p. 31:1–31:35. Consulté sur http://doi.acm.org/10.1145/2480741.2480748

SHAPLEY L. S. (1953). A value for n-person games. *Contributions to the Theory of Games*, n° 28, p. 307-317. Consulté sur https://ci.nii.ac.jp/naid/10013542751/en/

Štrumbelj E., Kononenko I. (2008). Towards a model independent method for explaining classification for individual instances. In I.-Y. Song, J. Eder, T. M. Nguyen (Eds.), *Data warehousing and knowledge discovery*, p. 273–282. Berlin, Heidelberg, Springer Berlin Heidelberg.

Strumbelj E., Kononenko I. (2010, mars). An efficient explanation of individual classifications using game theory. *J. Mach. Learn. Res.*, vol. 11, p. 1–18. Consulté sur http://dl.acm.org/citation.cfm?id=1756006.1756007

Vanschoren J., Rijn J. N. van, Bischl B., Torgo L. (2014, juin). Openml: Networked science in machine learning. *SIGKDD Explor. Newsl.*, vol. 15, n° 2, p. 49–60. Consulté sur http://doi.acm.org/10.1145/2641190.2641198

Wachter S., Mittelstadt B. D., Russell C. (2017). Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *CoRR*, vol. abs/1711.00399. Consulté sur http://arxiv.org/abs/1711.00399

Massively Distributed Dirichlet Process Mixture Models

Khadidja Meguelati [1], **Benedicte Fontez** [2], **Nadine Hilgert** [2], **Florent Masseglia** [1]

1. *Inria and LIRMM, Montpellier, France*
   firstname.lastname@inria.fr
2. *MISTEA, Univ. Montpellier, Montpellier SupAgro, INRA, Montpellier, France*
   benedicte.fontez@supagro.fr,nadine.hilgert@inra.fr

ABSTRACT. *Clustering with accurate results have become a topic of high interest. Dirichlet Process Mixture (DPM) is a model used for clustering with the advantage of discovering the number of clusters automatically. It is highly time consuming, which impairs its adoption and makes centralized DPM approaches inefficient. We propose DC-DPM (Distributed Clustering via DPM), a parallel clustering solution that gracefully scales to millions of data points while remaining DPM compliant, which is the challenge of distributing this process. Our experiments, on both synthetic and real life data, illustrate the high performance of our approach. The centralized algorithm does not scale and has its limit on 100K data points, where it needs more than 7 hours. In this case, our approach needs less than 30 seconds.*

RÉSUMÉ. *La classification non supervisée (ou clustering) a pour objectif d'identifier des classes pertinentes dans les données. Le mélange de processus de Dirichlet (DPM) est utilisé pour le clustering car il définit automatiquement le nombre de classes mais les temps de calculs qui l'impliquent sont généralement trop importants, nuisant à son adoption et rendant inefficaces ses versions centralisées. Dans la logique du DPM, nous proposons DC-DPM, une version parallélisée, qui s'adapte à des millions de points de données, ce qui représente un vrai défi. Nos expérimentations, tant sur des données synthétiques que réelles, illustrent la performance de notre approche. Comparativement, l'algorithme centralisé ne passe pas à l'échelle. Son temps de réponse est de plus de 7 heures sur des données de 100K points, quand notre approche prend moins de 30 secondes.*

KEYWORDS: *Dirichlet Process Mixture Model, Clustering, Parallelism*

MOTS-CLÉS : *Modèle de mélange de processus de Dirichlet, Classification non supervisée, parallélisme*

Distributed Clustering via DPM

One of the main difficulties, for clustering, is the fact that we don't know, in advance, the number of clusters to be discovered. In (Meguelati *et al*.,) we focus on the DPM approach since it allows estimating the number of clusters and assigning observations to clusters, in the same process. Unfortunately, DPM is highly time consuming. Consequently, several attempts have been done to make it distributed. However, while being effectively distributed, these approaches usually suffer from convergence issues (imbalanced data distribution on computing nodes) (Lovell *et al*.,) or do not fully benefit from DPM properties (Wang, Lin,). Furthermore, making DPM parallel is not straightforward since it must compare each record to the set of existing clusters, a highly repeated number of times. That impairs the global performances of the approach in parallel, since comparing all the records to all the clusters would call for a high number of communications and make the process impracticable. We propose DC-DPM (Distributed Clustering by Dirichlet Process Mixture), a distributed DPM algorithm that allows each node to have a view on the local results of all the other nodes, while avoiding exhaustive data exchanges. The main novelty of our work is to propose a model and its estimation at the master level by exploiting the sufficient statistics from the workers, in a DPM compliant approach. Our solution takes advantage of the computing power of distributed systems by using parallel frameworks such as MapReduce or Spark (Zaharia *et al*.,). Our DC-DPM solution distributes the Dirichlet Process by identifying local clusters on the workers and synchronizing these clusters on the master. These clusters are then communicated as a basis among workers for local clustering consistency. We modified the Dirichlet Process to consider this basis in each worker. By iterating this process we seek global consistency of DPM in a distributed environment. Our experiments, using real and synthetic datasets, illustrate both the high efficiency and linear scalability of our approach. We report significant gains in response time, compared to centralized DPM approaches, with processing times of a few minutes, compared to several days in the centralized case.

Acknowledgements: The research leading to these results has received funds from the European Union's Horizon 2020 Framework Programme for Research and Innovation, under grant agreement No. 732051.

References

Lovell D., Adams R. P. Mansingka V. (2012). Parallel markov chain monte carlo for dirichlet process mixtures. In *Workshop on big learning, nips.*

Meguelati K., Fontez B., Hilgert N. Masseglia F. (2019, April). Dirichlet process mixture models made scalable and effective by means of massive distribution. In *SAC: Symposium on Applied Computing.* Limassol, Cyprus. Retrieved from https://hal.archives-ouvertes.fr/hal-01999453

Wang R. Lin D. (2017). Scalable estimation of dirichlet process mixture models on distributed data. In *Proceedings of the 26th international joint conference on artificial intelligence*, pp. 4632–4639. AAAI Press. Retrieved from http://dl.acm.org/citation.cfm?id=3171837.3171935

Zaharia M., Chowdhury M., Franklin M. J., Shenker S. Stoica I. (2010). Spark: Cluster computing with working sets. In *Hotcloud.*

Lambda Architecture pour une analyse à haute performance des données des réseaux sociaux

Annabelle Gillet, Éric Leclercq, Nadine Cullot

Laboratoire d'Informatique de Bourgogne - EA 7534
Univ. Bourgogne Franche-Comté
9, Avenue Alain Savary
F-21078 Dijon - France
annabelle.gillet@depinfo.u-bourgogne.fr, prénom.nom@u-bourgogne.fr

RÉSUMÉ. *Dans cet article, nous montrons comment une Lambda Architecture contribue à l'élaboration d'une plateforme de collecte et d'analyse en temps réel de données de Twitter. Après avoir présenté le contexte, détaillé les besoins et identifié les spécificités attendues, nous comparons les architectures Lambda et Kappa et nous dressons un état de l'art de l'utilisation de la Lambda Architecture dans différents domaines. Nous proposons ensuite une adaptation de la Lambda Architecture pour permettre le stockage de données dans un polystore et pour tenir compte de la multitude des types d'analyse à appliquer pour répondre à des projets de recherche en sciences sociales et en sciences de la communication dont les objectifs sont d'étudier la structure de la communication et la circulation du discours sur Twitter autour de sujets de société.*

ABSTRACT. *In this article, we show how a Lambda Architecture can contribute to the development of a platform for collecting and analyzing, in real-time, data from Twitter. After having presented the context, detailed the needs and identified the expected specificities, we compare the Lambda and Kappa architectures and we describe the state of the art on Lambda Architecture use in different domains. We propose an adaptation of the Lambda architecture to allow the storage of data in a polystore and to take into account different types of analysis to be carried out to answer researches in social sciences and communication sciences. In these projects the objectives are to study the structure of communication and the circulation of the speech on Twitter about some societal topics.*

MOTS-CLÉS : *Lambda Architecture, flux, polystore, données sociales*

KEYWORDS: *Lambda Architecture, streaming, polystore, social data*

1. Introduction

Les réseaux sociaux sont une source importante de données pouvant être exploitées dans de nombreux domaines comme le marketing, la politique, les sciences sociales. Les données produites font partie des Big Data et elles nécessitent des architectures logicielles spécifiques capables de prendre en compte des flux importants, de les stocker et de les analyser (Kambatla *et al*., 2014 ; Singh, Reddy, 2015). Ces problématiques sont regroupées sous les termes *High Performance Data Analysis* (HPDA) et *Data Intensive High Performance Computing*. Les données des réseaux sociaux sont riches mais leur sémantique est complexe et elles contiennent de nombreuses caractéristiques latentes. Pour effectuer une analyse détaillée de ces données, il est nécessaire d'avoir recours à plusieurs algorithmes comme la détection de communautés, la détection d'événements, la recherche d'utilisateurs influents, etc. Les résultats des analyses peuvent être ensuite exploités pour éclairer une question de recherche, élaborer une théorie, un modèle qui seront ensuite validés sur d'autres données sociales.

En fonction des observations que l'on souhaite réaliser ou des questions auxquelles on cherche à répondre, plusieurs algorithmes sont mis en œuvre et ils travaillent sur différentes modélisations des données. Par exemple, on peut vouloir trouver des variations des comportements des utilisateurs au fil du temps, auquel cas les séries temporelles sont bien adaptées pour représenter les données. Ensuite, sur une période identifiée, on peut regrouper des utilisateurs en communautés, afin d'avoir une vision plus précise de l'organisation et des interactions entre les communautés d'utilisateurs. Ainsi, plusieurs algorithmes sont appliqués séparément ou conjointement et ils fournissent des informations à différents niveaux de granularité (macroscopique, mésoscopique et microscopique). De plus, ils peuvent être exécutés en temps réel sur un flux ou en temps différé sur une plus grande quantité de données. Dans cet article, nous traitons plus particulièrement du patron Lambda Architecture (Marz, 2011) intégrant un traitement des flux en temps réel et des analyses en temps différé. Nous spécialisons l'architecture pour une plateforme de collecte et d'analyse de données de Twitter. Nous testons l'architecture avec différents jeux de données précédemment collectés.

L'article est organisé comme suit, la section 2 présente le contexte et décrit les principes des architectures de traitement de données à haute performance. La section 3 est un état de l'art sur l'utilisation des Lambda Architectures, la section 4 détaille notre proposition et la section 5 présente les résultats d'une série d'expériences permettant de valider les différents composants de l'architecture. La section 6 conclut l'article et dresse les perspectives dégagées par ce travail.

2. Contexte et problématique

Nous nous plaçons dans le cadre de projets interdisciplinaires développés avec des chercheurs en sciences sociales et en sciences de la communication dont l'objectif est d'étudier la structure de la communication et la circulation des discours sur Twitter (Basaille *et al*., 2017). Les thématiques abordées sont les questions de société comme les problèmes environnementaux, le changement climatique ainsi que les discours au-

tour de l'alimentation intégrant les comportements alimentaires, les crises sanitaires, etc. Dans ces domaines, les chercheurs en sciences sociales souhaitent détecter des messages viraux, l'émergence de hashtags populaires ou de topics c'est-à-dire des groupes de hashtags utilisés ensemble et structurant les discours. Il veulent aussi obtenir des informations sur les communautés d'utilisateurs ayant des comportements similaires et suivre leur évolution.

Afin d'effectuer des analyses, il faut tout d'abord pouvoir collecter les données et être capable d'absorber un flux important [1] de plusieurs milliers de messages par seconde pour constituer un jeu de données se rapprochant le plus possible de la réalité. Twitter met à disposition des développeurs des moyens pour récupérer les données publiées par leurs utilisateurs, à travers deux API différentes : *search* et *stream*. L'API *stream* est limitée à 1% du trafic global par machine connectée. Ce volume peut être très important si plusieurs machines sont utilisées en parallèle (500.10^6 tweets sont publiés quotidiennement). De plus, les tweets contiennent de nombreuses informations qu'il faut extraire comme l'utilisateur émetteur, la date d'émission, les hashtags utilisés, les utilisateurs mentionnés, la localisation, les URL citées, etc. En conséquence, une plateforme d'analyse à haute performance doit être capable d'absorber, en temps réel, un flux important de tweets, d'effectuer des analyses elles aussi en temps réel et de stocker les données en vue d'analyses plus complexes réalisées en temps différé. Avant de pouvoir utiliser les données collectées, il faut bien souvent les nettoyer, transformer les données brutes dans un modèle plus adapté aux algorithmes utilisés. Afin de limiter l'utilisation des processus ETL (Extract-Load-Transform) très consommateurs en temps de développement, les *polystores* proposent différents modes de stockage unifiés soit par le langage, soit par un modèle commun (Leclercq, Savonnet, 2018). Ils prennent ainsi en compte des données de type graphes attributaires, matricielles, relationelles et documents [2]. Par ailleurs, dans les domaines d'application qui nous intéressent, il est nécessaire de pouvoir suivre en permanence l'évolution des tendances, de détecter des événements. Ces analyses en temps réel peuvent être réalisées par des algorithmes de *streaming* (Gama, 2012 ; Silva *et al*., 2013 ; McGregor, 2014) nommés algorithmes de fouille de flux de données; en ligne ou encore incrémentaux.

2.1. Principe de la Lambda Architecture

La Lambda Architecture est un patron d'architecture logicielle décrit par Nathan Marz (Marz, 2011), qu'il détaillera plus tard dans un ouvrage plus complet (Marz, Warren, 2015). Cette architecture permet de traiter les données massives en temps réel et par lots de manière simultanée. Les trois V d'origine des Big Data concernant le Volume, la Variété et la Vélocité ont été complétés par IBM avec la Véracité puis ensuite

1. Le flux moyen moyen mondial est de 6 000 tweets environ par seconde, avec de rares pics dépassant plus de 8 000 tweets par seconde sur des événements médiatisés comme les MTV Video Music awards (source http://www.internetlivestats.com/twitter-statistics/)

2. Utiles par exemple pour stocker le contenu des tweets et le contenu Web des URL référencées dans les tweets.

par SAS qui a introduit la Variabilité et enfin Oracle qui a ajouté la Valeur (Gandomi, Haider, 2015). La Lambda Architecture se concentre sur quelques V caractéristiques, la Vélocité, le Volume et dans une moindre mesure sur la Valeur. L'architecture se compose de trois couches (ou *layers*) : la *Batch layer*, la *Serving layer* et la *Speed layer* (figure 1). Le principe général est le suivant : la *Batch layer* permet de fournir une vue exacte sur les données, mais avec un temps de traitement ou de transformation important. La *Serving layer* met ensuite ces vues à disposition des utilisateurs, avec pour objectif d'optimiser le temps d'accès aux données. La *Speed layer* sert à compenser ce temps de traitement, en proposant un traitement des données en temps réel.

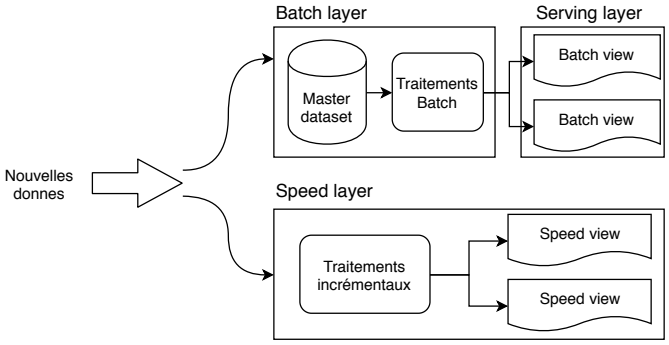


Figure 1. Principe général de la Lambda Architecture

La *Batch layer* a deux rôles principaux : 1) stocker les nouvelles données, et 2) réaliser des calculs, des transformations de modèles sur ces données. Le stockage des données brutes ne se fait pas sous la forme d'états mais sous la forme d'une suite d'événements ou de messages qui permettent d'obtenir un état actualisé des différents éléments lorsque ces évènements sont traités dans leur ensemble. La *Batch layer* stocke toutes les données (messages) qu'elle reçoit dans le *master dataset*, c'est-à-dire un entrepôt permanent de données brutes. Les calculs et les transformations de modèles sont opérés à partir du *master dataset*, souvent de manière distribuée afin de garantir le passage à l'échelle.

La *Serving layer* se charge ensuite de récupérer les résultats de ces traitements, et les met à la disposition des utilisateurs. Le but étant d'optimiser le temps d'accès, la modélisation des données est réalisée dans ce sens (utilisation des index, schéma non normalisé, etc.).

La *Speed layer* s'occupe de traiter les données en temps réel, au moyen d'algorithmes qui travaillent de manière incrémentale. Elle rend disponible les nouvelles données qui n'ont pas encore été traitées par la *Batch layer*. La performance prime donc sur l'exactitude (ou la véracité) des traitements réalisés, cette contrainte étant laissée à la charge de la *Batch layer*.

Les erreurs d'exactitude qui peuvent survenir lors du traitement de la *Speed layer* sont corrigées lorsque la *Batch layer* termine son traitement sur les mêmes données. Réciproquement, la lenteur de la *Batch layer* est bien compensée par la rapidité de la *Speed layer*, qui permet de voir un résultat des traitements en temps réel mais éventuellement avec des imprécisions.

La force de la Lambda Architecture vient donc de sa capacité à traiter des données en quantité importante et avec un flux soutenu, en compensant la latence des traitements *Batch* par des traitements sur des flux par la *Speed Layer*. Toutefois, notre domaine d'application induit une variation dans la construction de la Lambda Architecture. Nous devons satisfaire deux priorités : 1) préparer les données pour des analyses dont les algorithmes ne sont pas nécessairement connus *a priori* et 2) pouvoir calculer en temps réel des indicateurs macroscopiques. Dans la littérature la Lambda Architecture est critiquée car les traitements sont dupliqués dans les couches *Batch* et *Speed*, mais en réalité ce n'est pas valable pour tous les domaines d'application et encore moins pour notre cas d'utilisation.

2.2. Lambda, Kappa, et autres architectures, discussion

La Kappa Architecture est un autre patron qui adopte une vision différente de la Lambda Architecture. Elle peut être vue comme une simplification dans le sens ou elle considère que tous les traitements opèrent sur des flux (*everything is a stream*). Comme (Kreps, 2014) le décrit dans son article, plutôt que d'avoir une couche *Batch* et une couche *Speed*, il est possible de conserver uniquement une seule couche *Speed* et d'organiser les flux. Pour ce faire, les données doivent être conservées sous la forme d'une suite de messages (*logs*). S'il faut relancer un traitement sur les données, il suffit d'exécuter un autre traitement *Speed* en parallèle du principal, qui contient les modifications à apporter, et d'arrêter le premier lorsque le nouveau l'a rattrapé. Dans ce contexte Apache Kafka [3] est une solution utilisée majoritairement dans des architectures avec une composante temps réel. Bien que la durée de stockage (rétention) ou le volume des données soient limités, il est tout même possible de traiter les données depuis le début en les stockant en plus dans un système de fichiers distribué comme Apache Hadoop HDFS [4], pour ensuite les réinjecter dans des files de messages.

À titre de comparaison, les points forts de la Lambda Architecture sont : une conservation des données brutes qui permet de les retraiter si besoin, une solution flexible indépendante des technologies et une adaptation fine des outils d'analyse aux demandes des clients. Il s'agit cependant d'une architecture assez complexe à mettre en œuvre. Les points forts de la Kappa Architecture sont l'unification du principe de traitement par flux, sa simplicité et sa relative indépendance vis-à-vis des technologies. Cependant la Kappa Architecture n'est pas aussi flexible que la Lambda Architecture du point de vue de l'organisation des données. L'architecture SMACK (*Spark-Mesos-*

3. https://kafka.apache.org/
4. https://hadoop.apache.org/

Akka-Cassandra-Kafka) est une autre proposition d'architecture, figée du point de vue des outils, même si elle est souvent présentée comme concurrente des Kappa et Lambda Architecture, il s'agit plutôt d'une pile de technologies (*data analytics stack*).

En conclusion, la flexibilité de la Lambda Architecture doit être mise en perspective du haut niveau de technicité requis pour la mise en œuvre des différentes solutions techniques qu'elle intègre. Dans notre cas, il s'agit de la solution la plus pertinente au regard des différents types d'analyses à déployer sur des graphes, des séries temporelles, des données relationnelles ou des données textuelles. Une proposition dans ce sens a été élaborée dans l'article (Fernandez *et al*., 2015) pour LinkedIn avec l'architecture Liquid dont le principe fondateur est un chemin à faible latence (*Low Latency Path*). L'architecture Liquid recueille moins d'attention que la Lambda Architecture. En effet, cette dernière insiste sur une capacité d'analyse temps réel très séduisante alors que le patron d'architecture est très général et délègue aux composants logiciels retenus pour l'implémentation le soin d'assurer la prise en charge des flux et des traitements.

3. État de l'art

Avant de présenter quelques exemples de projets appliqués, nous décrivons plusieurs travaux qui permettent d'identifier les spécificités des Lambda Architectures.

Dans (Mishne *et al*., 2013) les auteurs relatent d'expériences d'architecture pour la correction orthographique et la suggestion de recherche en temps réel de tweets. Ces travaux montrent les limites des architectures s'appuyant uniquement sur Hadoop et ils sont les précurseurs de la Lambda Architecture. L'article (Lin, 2017) met en avant les concessions qu'implique l'adoption d'une Lambda Architecture. En effet, le mode de fonctionnement de la *Speed layer* et de la *Batch layer* sont différents, mais ils doivent pourtant produire des résultats similaires. Cela entraîne une complexité lors du développement et des évolutions de l'architecture. Dans (Feick *et al*., 2018) les auteurs décrivent et comparent les architectures Lambda et Kappa par rapport au théorème CAP et présentent de manière synthétique des outils d'implémentation pour les différentes couches. Ils relatent d'une expérimentation avec des données de Twitter avec un flux maximum de 26 000 tweets par jour et 2 300 par heure.

Du point de vue des domaines d'applications, on distingue deux catégories de projets : des instanciations de la Lambda Architecture pour des domaines précis et des travaux autour de la construction de plateformes génériques par assemblage de solutions logicielles open source.

Dans (Munshi, Mohamed, 2018), les auteurs présentent une implémentation de la Lambda Architecture appliquée aux traitements des données issues des réseaux électriques. La résistance aux pannes, le temps de réponse rapide, le passage à l'échelle et la flexibilité de ce type d'architecture sont les paramètres qui ont motivés leur choix d'utiliser la Lambda Architecture. Hadoop HDFS est utilisé pour implanter un *Data-Lake* et traiter de la diversité des données (capteurs, images, vidéos). La couche

d'interrogation, séparée de la couche d'analyse, intègre les technologies SparkSQL [5], Hive [6] et Impala [7]. Lee et Lin (2017) spécifient une Lambda Architecture pour construire un système de recommandation de restaurants. Leur apport est l'utilisation d'Apache Mesos [8] pour abstraire les composants matériels de la plateforme, faciliter son déploiement et sa mise à l'échelle. Les technologies utilisées sont Kafka pour la *Speed Layer* et les traitements temps réel, Hadoop HDFS pour le stockage des données brutes, et Spark pour la *Batch Layer*. Les performances de l'architecture sont testées avec le jeu de données MoviesLens 20M [9] et différentes configurations matérielles allant jusqu'à exploiter 32 coeurs CPU et 96Go RAM. Cependant, le volume des données et les flux sont trop peu importants pour fournir des expériences significatives.

RADStack (Yang *et al*., 2017) est une plateforme *open source* pour produire des analyses interactives, implémentant le principe de la Lambda Architecture et utilisant les briques logicielles Kafka, Samza [10], Hadoop et Druid [11]. Kiran *et al*. (2015) s'intéressent aux performances que peut offrir la Lambda Architecture, afin de minimiser les coûts qui peuvent être associés au déploiement des applications dans le *cloud*. Leur approche est mise en pratique avec le déploiement de leur architecture sur Amazon AWS, afin d'analyser des données issues de capteurs. Pal *et al*. (2018) proposent une évolution de la Lambda Architecture en intégrant un système multi-agents pour des applications de commerce électronique.

Comme Liquid (Fernandez *et al*., 2015) discutée précédemment, le système S-Store (Meehan *et al*., 2016) partage des similarités avec notre proposition en intégrant le polystore BigDAWG. Le rôle de S-Store est de nettoyer les données et de les transformer en vue de leur ingestion par le polystore. L'article présente un *proof-of-concept* avec des données des benchmarks TCP-C et TCP-DI [12]. Plutôt que d'écrire ses résultats dans un fichier, puis d'insérer le contenu du fichier en base de données comme un ETL standard, S-Store fait passer les données de processus en processus afin de leur appliquer une transformation jusqu'à leur insertion en base de données. L'intégration dans une Lambda Architecture est évoquée dans les perspectives mais dans l'état actuel la proposition est plutôt un ETL traitant des flux en temps réel.

4. L'architecture Hydre

L'architecture Hydre que nous proposons s'appuie sur la Lamba Architecture et profite de ses avantages, tout en l'adaptant afin de mieux répondre aux besoins de notre contexte de collecte, gestion et analyse des données issues de Twitter.

---

5. https://spark.apache.org/sql/
6. https://hive.apache.org/
7. https://impala.apache.org/
8. http://mesos.apache.org/
9. https://grouplens.org/datasets/movielens/20m/
10. http://samza.apache.org/
11. http://druid.io/
12. http://www.tpc.org/

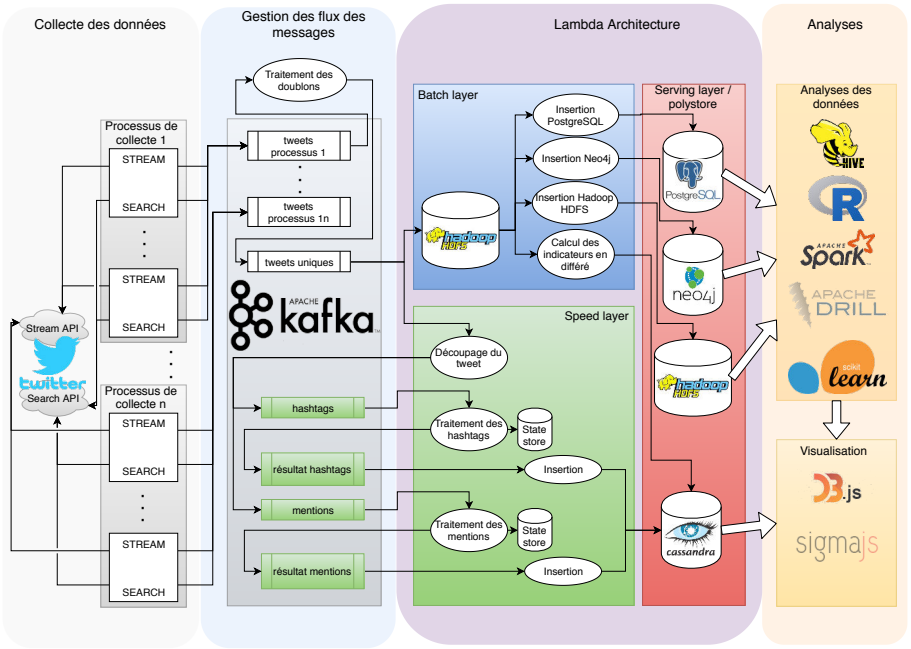


Figure 2. Hydre, Lambda Architecture détaillée

4.1. Architecture générale

La figure 2 présente l'architecture Hydre globale. Elle est majoritairement développée à l'aide du langage Scala [13], et agrège différents composants : 1) un système de collecte des données, faisant appel aux API de Twitter pour alimenter 2) un composant de gestion des flux des messages, qui utilise Kafka. L'extraction des messages ainsi collectés constitue notre point d'entrée dans la Lambda Architecture, et cette extraction est donc réalisée en parallèle par les deux couches : 3) la *Speed layer*, qui nous permet de calculer les séries temporelles sur les éléments importants par tranches d'une heure, comme des hashtags, des mentions, etc., et 4) la *Batch layer* qui enregistre d'abord les données brutes dans le *master dataset* implémenté avec Hadoop HDFS, puis réalise différents traitements sur ces données (comme recalculer les séries temporelles), et la *Serving layer* se charge d'insérer les données dans le stockage polystore. Ces données pourront ensuite être analysées et visualisées à l'aide de différents outils.

13. https://www.scala-lang.org/

4.2. La collecte des données

Afin de collecter des tweets, Twitter met à disposition deux API : l'API *search* [14] et l'API *stream* [15]. Bien que le but de ces API soit le même, leur fonctionnement est très différent. En effet, la première permet de collecter des tweets ayant été produits au maximum 7 jours plus tôt et qui correspondent à la requête qui a été fournie à l'API, tandis que la deuxième permet d'enregistrer une requête auprès de Twitter, qui renverra ensuite tous les tweets nouvellement produits correspondant aux critères spécifiés. Les critères des requêtes permettent de filtrer les tweets sur des mots-clés, des hashtags, des utilisateurs, etc. Les deux API nous fournissent les tweets au format JSON. Dans leur version gratuite, ces API imposent toutefois quelques limitations par compte utilisé et par adresse IP. En ce qui concerne l'API *search*, il est seulement possible d'envoyer 180 requêtes par créneaux de 15 minutes, en récupérant 100 tweets maximum à chaque fois. L'API *stream*, quant à elle, limite le nombre de tweets captés par l'application à 1% du trafic total de Twitter par machine connectée, et si les tweets ne sont pas consommés assez rapidement par l'application, la connexion avec Twitter peut être coupée. Pour communiquer avec ces API, nous utilisons la librairie Twitter4j [16].

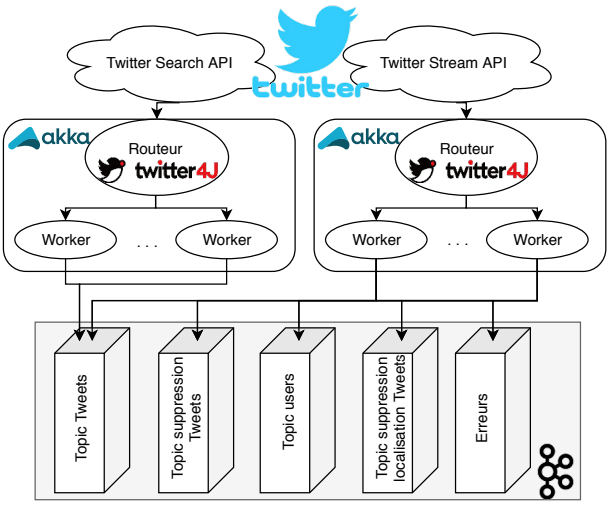


Figure 3. Collecte de tweets et articulation avec Kafka

Pour gérer les tweets que nous récoltons, nous utilisons des acteurs Akka [17] disponibles avec Scala. Akka est une implémentation de l'*Actor Model*, qui permet de

14. https://developer.twitter.com/en/docs/tweets/search/overview
15. https://developer.twitter.com/en/docs/tweets/filter-realtime/overview
16. http://twitter4j.org/en/index.html
17. https://akka.io/

réaliser des applications distribuées à base d'échanges de messages. Cela nous permet de paralléliser le traitement des tweets, de produire des messages de type (K, V) où K est la clé et V la valeur (le contenu du tweet). Si la rapidité n'est pas un critère primordial pour l'API *search* aux vues de ses limitations, cela devient un problème plus contraignant pour l'API *stream* dans le cas où le flux capté est très important. Nous utilisons la fonctionnalité *Routing* d'Akka, qui permet de définir un certain nombre de *Workers* auxquels le routeur transmet les messages qu'il reçoit pour qu'ils puissent les traiter (figure 3). Le nombre de *Workers* peut être fixe, comme c'est le cas pour notre traitement à la réception des tweets provenant de l'API *search*, ou il peut être adaptable dynamiquement au volume de messages que le routeur reçoit, ce que nous utilisons pour traiter les tweets provenant de l'API *stream*. L'architecture permet de traiter plusieurs corpus simultanément, chaque corpus dispose d'un ou plusieurs mécanismes de collecte en fonction de la volumétrie de tweets attendue pour le corpus (figure 2, partie grisée).

4.3. Système de gestion des flux

Le système Kafka est utilisé pour servir d'intermédiaire entre la partie collecte et les couches *Speed* et *Batch*. Il permet de créer des *topics* dans lesquels les producteurs peuvent poster des messages. Mais il permet aussi de former plusieurs groupes de consommateurs qui consultent les messages des *topics* à leur rythme et de manière indépendante (figure 2). Les messages sont alors conservés pendant un temps défini (délai de rétention) ou jusqu'à ce qu'ils atteignent un certain volume. Chaque topic a un usage particulier : gestion des messages tweets, profils utilisateurs, indicateurs de suppression, messages d'erreur.

Étant donné la configuration de la collecte, il est possible de recevoir un même tweet plusieurs fois. Que ce soit parce qu'un même mot-clé nous a permis de récupérer le tweet avec l'API *stream* puis avec l'API *search*, ou parce que des mots-clés différents utilisés sur des machines distinctes ciblent un même tweet. Bien que Kafka propose un mécanisme de *log compaction*, c'est-à-dire que pour une même clé de message seul le dernier message est conservé, cela n'est pas adapté à notre situation. En effet, si l'on prend en compte un tweet récupéré par l'API *stream*, et le même tweet qui peut être récupéré par l'API *search* jusqu'à 7 jours plus tard, il est fortement probable que le premier tweet ait déjà été traité avant qu'il ne soit remplacé par le deuxième, ce qui entraînerait un traitement en doublon. Nous développons donc un mécanisme spécifique pour le traitement des doublons en sortie de Kafka à l'aide de Kafka Streams. À la suite de ce traitement, les tweets uniques sont redirigés dans un topic, dans lequel les processus *Speed* et *Batch* vont puiser leurs données. Grâce à ce fonctionnement, nous pouvons alimenter les parties *Speed* et *Batch* avec des données uniques, et nous pouvons également récupérer des statistiques sur la collecte qui nous permettent de savoir combien de fois un tweet a été collecté, depuis quelle API et avec quelle requête. Ces éléments pourraient ensuite être utilisés dans une évolution de l'architecture, afin d'optimiser la répartition des mots-clés dans les requêtes utilisées par les machines de collecte pour grouper ceux qui apparaissent souvent ensemble.

4.4. La couche Speed

Le rôle principal de la *Speed layer* est, dans notre cas, de produire des indicateurs macroscopiques sur une collecte en cours et détecter des événements, comme les hashtags les plus utilisés, leur fréquence, ou les mentions faites d'un utilisateur. Ces indicateurs prennent la forme de séries temporelles par tranches de temps d'une heure. Cette couche est développée à l'aide de Kafka Streams, et les résultats sont insérés dans une base de données Cassandra.

Le fonctionnement est le suivant (figure 2 cadre et rectangles verts), un tweet est d'abord traité par un processus qui se charge de séparer les différentes parties (les hashtags, les mentions, etc.). Ces parties sont envoyées dans différents topics Kafka dans le but d'être traitées chacune par un processus spécifique. Les processus sont associés à un *state store* de Kafka, qui leur permettent de garder un état en mémoire. Nous nous en servons pour compter combien de fois un même élément a été rencontré sur une plage de temps donnée. Le compte mis à jour est ensuite envoyé à nouveau dans un topic, puis réceptionné par Cassandra qui se charge de mettre les données à jour.

Séparer les traitements de cette manière autorise une grande souplesse dans les évolutions de la couche *Speed*. En effet, il suffit juste d'adapter le processus de découpage pour extraire les nouveaux éléments qui pourraient nous intéresser, et chaque processus de traitement étant indépendant, il peut être ajouté, modifié ou retiré sans impacter le reste de la couche.

4.5. La couche Batch et la couche Serving

Lorsque de nouveaux tweets sont collectés et envoyés via Kafka, le processus *Batch* les récupère pour les ajouter dans le *master dataset*. Afin de remplir cette tâche, nous utilisons Hadoop HDFS, qui est un framework permettant de distribuer de larges volumes de données sur un cluster et d'y effectuer des traitements en utilisant le paradigme *map-reduce*. L'ajout de machines au cluster permet un passage à l'échelle efficace, en augmentant à la fois la capacité de calcul et de stockage. De nombreux autres outils constituent cet eco-système et permettent d'abstraire les mécanismes proposés par Hadoop.

Deux types de processus sont développés. Un processus *Batch* se charge de recalculer les séries temporelles produites par le processus *Speed* afin de les remplacer. Un ensemble de processus est chargé de récupérer les nouvelles données et, pour chaque base de données composant notre polystore, de les insérer tout en respectant le schéma défini. De cette manière, les données que nous récoltons sont transformées dans différents modèles afin de pouvoir ensuite être exploitées par les différents algorithmes d'analyse. Différents types de bases de données sont utilisés afin de correspondre aux besoins des algorithmes : Cassandra pour les séries temporelles, Noe4j pour les modèles graphes et PostgreSQL pour stocker les données des tweets comme la géolocalisation ou les informations des utilisateurs. Grâce à l'organisation des éléments

de notre architecture, il est aisé de rajouter une autre base de données, de modifier ou ajouter un schéma d'insertion si cela est nécessaire. Il suffit en effet de modifier ou de rajouter un processus d'insertion dans les parties *Batch* et *Serving* afin que ces changements soient pris en compte.

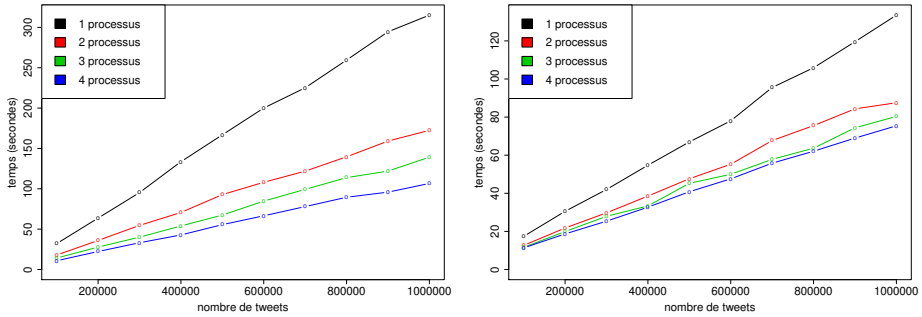
5. Retours d'expériences

Afin de valider la capacité de l'architecture Hydre à supporter la charge imposée à ses composants, différentes expériences sont réalisées. Elles sont définies pour les 3 niveaux de l'architecture : 1) au niveau du système de production des messages, pour mesurer la capacité de Kafka à ingérer des flux importants, 2) au niveau de la couche *Batch*, de la récupération des messages depuis Kafka pour enregistrer les tweets dans HDFS, à l'extraction des données d'HDFS pour les insérer dans les systèmes de stockage du polystore, 3) au niveau de la *Speed layer* pour mesurer sa capacité à produire des séries temporelles et à les matérialiser dans Cassandra. La configuration matérielle sur laquelle ont été effectués les tests est la suivante : 1) un serveur Dell R710 (Intel Xeon CPU E5-2650 v2 @ 2.60GHz, 16 cœurs, 128Go RAM) pour la couche de collecte et Kafka (3 brokers dans des conteneurs Docker) ; 2) un serveur Dell R940 (Intel(R) Xeon(R) CPU E7-4820 v2 @ 2.00GHz, 32 cœurs, 256Go RAM) pour les couches *Batch*, *Speed* et *Serving* ; 3) un cluster Hadoop 20 nœuds *data nodes* et 2 *name nodes*, 184 cœurs, 1040Go RAM, 70To de stockage HDFS.

Ingestion de Kafka des messages. Kafka est un composant principal de l'architecture. Il garde tous les tweets obtenus avec les processus de collecte pour qu'ils puissent être consommés par les couches *Speed* et *Batch*. Afin d'évaluer la capacité de cette partie à supporter un flux important, des producteurs Kafka envoient un certain nombre de tweets, et le temps est mesuré depuis l'envoi du premier message jusqu'à la réception du dernier. Pour réduire l'impact des éléments extérieurs sur l'expérience, les tweets sont obtenus à partir d'un modèle, et seul l'identifiant de chaque tweet est produit aléatoirement (figure 4a).

Enregistrement des tweets dans HDFS. Dans la partie *Batch*, le premier évènement lors de l'arrivée d'un message est son enregistrement sous forme de donnée brute dans Hadoop. Nous produisons donc en amont un certain nombre de messages uniques dans Kafka, que nous consommons ensuite. Nous mesurons le temps pris pour que le nombre de tweets défini soit enregistré dans Hadoop (figure 4b).

Insertion dans un schéma relationnel normalisé ou non. Une fois que les tweets sont enregistrés dans Hadoop, un consommateur de la partie *Batch* peut ensuite les récupérer pour les enregistrer dans une base de données. C'est ce que nous testons dans cette expérience, en extrayant les informations du tweet de son format JSON et en l'insérant dans une base de données PostgreSQL sous une forme non normalisée (une seule table) (figure 5a) et normalisée (11 tables avec index) (figure 5b). Nous réalisons aussi l'expérience pour l'insertion dans Neo4j (3 types de noeuds et 4 types de relations) (figure 5c). Pour ces trois cas, nous mesurons le temps que le processus met pour extraire les tweets d'Hadoop et les insérer en base de données.



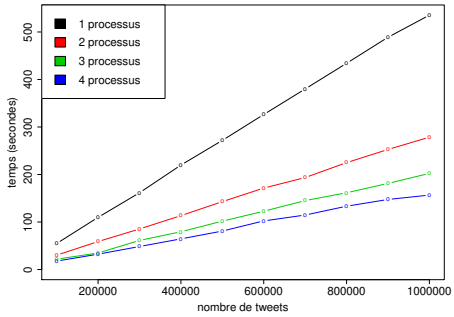
(a) Ingestion dans Kafka (b) Consommation et enregistrement HDFS

Figure 4. Production des messages (Akka, Kafka) et enregistrement (Kafka, HDFS)

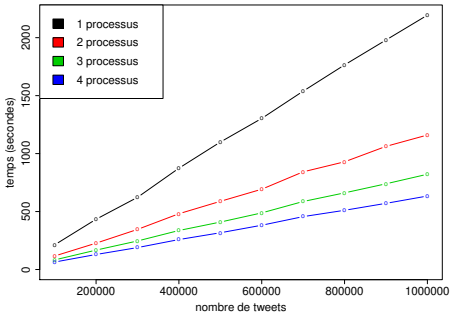
Mise à jour de séries temporelles. Cette expérience nous sert à mesurer le temps nécessaire à la couche *Speed* pour mettre à jour la série temporelle concernant les hashtags des tweets. Le traitement se compose de 3 étapes réalisées avec Kafka Streams : 1) extraire les hashtags du tweet, 2) compter le nombre d'occurrences par heure, 3) insérer ce résultat dans Cassandra. Pour cette expérience, les messages Kafka sont d'abord produits, puis le traitement *Speed* est lancé. Le temps est mesuré depuis l'extraction des hashtags du premier message, jusqu'à la dernière insertion dans Cassandra (figure 6a). La même expérience est réalisée pour mettre à jour les séries temporelles, mais cette fois du côté de la partie *Batch* : les données sont extraites d'Hadoop, une requête est envoyée sur Cassandra pour obtenir le nombre d'occurrences pour le hashtag en train d'être traité, puis ce nombre d'occurrences est incrémenté (figure 6b).

L'étude des performances des composants principaux d'Hydre est synthétisée dans le tableau suivant qui donne le nombre de tweets par seconde traitable par chaque composant et démontre la capacité de l'architecture à supporter le flux moyen de Twitter.

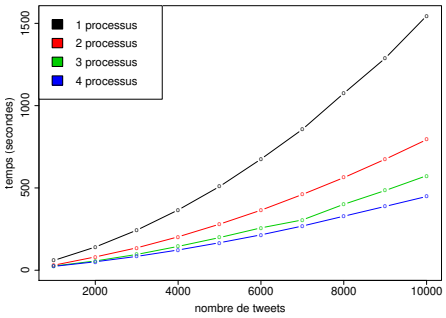
Interprétation des résultats. On observe que les différents composants traitent les flux de tweets d'une manière plutôt linéaire, et se prêtent assez bien à la parallélisation (figures 4, 5 et 6). Les performances de la partie *Speed* restent très bonnes par rapport à celles de la partie *Batch* pour ce qui concerne le calcul des séries temporelles (d'autant plus qu'il faut compter le temps d'enregistrement dans Hadoop pour la partie *Batch*). Neo4j, quant à lui, présente d'importants problèmes de performances, et même s'il se parallélise bien, il montre une allure exponentielle lors d'opérations d'insertion. Cela nous amène à étudier d'autres bases de données orientées graphes afin de le remplacer par un système plus adapté à nos besoins. Nous indiquions plus haut que le flux moyen de Twitter est de 6 000 tweets/s et que nous récupérions au maximum 1% de ce flux par machine, les résultats que nous avons obtenus (tableau 1) nous permettent de traiter sereinement les quantités de tweets que nous nous attendons à recevoir.



(a) Insertion dans PostgreSQL (schéma non normalisé)



(b) Insertion dans PostgreSQL (schéma normalisé)



(c) Insertion dans Neo4j

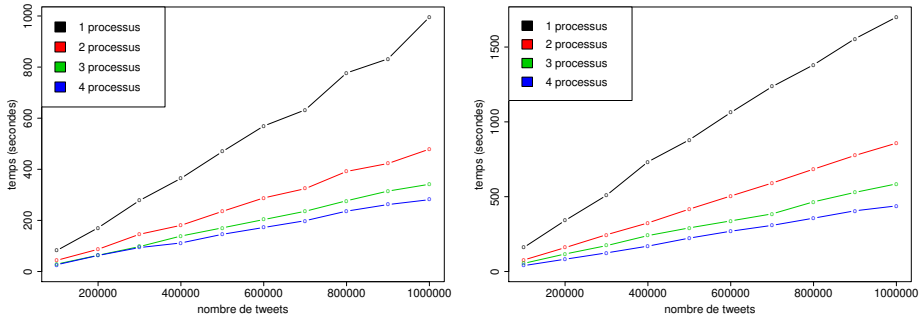
Figure 5. Lecture depuis Hadoop HDFS et insertion dans PostgreSQL et Neo4j

Tableau 1. Nombre de tweets par seconde admissible pour chaque composant

Nombre de processus	Ingestion Kafka	Stockage HDFS	Insertion non normalisé	Insertion normalisé	Insertion Neo4j	Série temporelle (Speed)	Série temporelle (Batch)
1	3 082	7 181	1 840	460	10	1 091	579
2	5 615	10 194	3 508	855	19	2 149	1208
3	7 227	11 364	4 977	1 210	26	3 010	1739
4	9 128	12 017	6 089	1 563	30	3 459	2324

6. Conclusion

Nous avons présenté une adaptation de la Lambda Architecture afin de répondre aux spécificités des analyses de données pour des projets en sciences sociales pour



(a) Production de la série temporelle hashtags dans Cassandra (couche *Speed*)

(b) Production de la série temporelle hashtags dans Cassandra (partie *Batch*)

Figure 6. Production de séries temporelles

lesquels les algorithmes à appliquer ne sont pas nécessairement connus *a priori*. Les composants sont architecturés autour de Kafka et permettent d'insérer des données dans un polystore et de calculer des indicateurs en temps réel pour détecter des événements. Nous avons ainsi développé et testé un bus de messages à faible latence, capable de supporter le flux des tweets produits quotidiennement. Notre implémentation permet de rajouter ou modifier simplement des traitements dans sa partie *Batch* ou *Speed*. Nous prévoyons d'améliorer les services de la couche *Speed* afin de détecter en temps réel des précurseurs d'évènements, des messages viraux et des robots. Sur un plan plus théorique nous allons réaliser une typologie des cas d'utilisation des données sociales pour les analyses, tester les performances des systèmes de stockage et intégrer le modèle TDM (Leclercq, Savonnet, 2018). Sur le plan technique, nous envisageons d'utiliser Kafka schema registry pour gérer les évolutions des schémas des messages, KSQL et/ou Apache Samza pour implanter les autres algorithmes de flux (Noghabi *et al.*, 2017).

Bibliographie

Basaille I., Kirgizov S., Leclercq É., Savonnet M., Cullot N., Grison T. *et al.* (2017). Un observatoire pour la modélisation et l'analyse des réseaux multi-relationnels. *Document numérique*, vol. 20, n° 1, p. 101–135.

Feick M., Kleer N., Kohn M. (2018). Fundamentals of Real-Time Data Processing Architectures Lambda and Kappa. In *Lecture Notes In Informatics (LNI)*, p. 1-12.

Fernandez R. C., Pietzuch P. R., Kreps J., Narkhede N., Rao J., Koshy J. *et al.* (2015). Liquid: Unifying Nearline and Offline Big Data Integration. In *Conference on Innovative Data System Research (CIDR'15)*.

Gama J. (2012). A survey on learning from data streams: current and future trends. *Progress in Artificial Intelligence*, vol. 1, n° 1, p. 45–55.

Gandomi A., Haider M. (2015). Beyond the hype: Big Data concepts, methods, and analytics. *International Journal of Information Management*, vol. 35, n° 2, p. 137–144.

Kambatla K., Kollias G., Kumar V., Grama A. (2014). Trends in Big Data Analytics. *Journal of Parallel and Distributed Computing*, vol. 74, n° 7, p. 2561–2573.

Kiran M., Murphy P., Monga I., Dugan J., Baveja S. S. (2015). Lambda architecture for cost-effective batch and speed big data processing. In *IEEE International Conference on Big Data*, p. 2785–2792.

Kreps J. (2014). Questioning the Lambda Architecture. *O'Reilly RADAR,online article, July*. Consulté sur https://www.oreilly.com/ideas/questioning-the-lambda-architecture

Leclercq É., Savonnet M. (2018). A tensor based data model for polystore: An application to social networks data. In *Proceedings of the 22nd International Database Engineering & Applications Symposium (IDEAS)*, p. 110–118.

Lee C.-H., Lin C.-Y. (2017). Implementation of Lambda Architecture: A Restaurant Recommender System over Apache Mesos. In *IEEE 31st International Conference on Advanced Information Networking and Applications (AINA)*, p. 979–985.

Lin J. (2017). The Lambda and the Kappa. *IEEE Internet Computing*, vol. 21, n° 5, p. 60–66.

Marz N. (2011). *How to beat the cap theorem.* Consulté sur http://nathanmarz.com/blog/how-to-beat-the-cap-theorem.html

Marz N., Warren J. (2015). *Big Data: Principles and best practices of scalable real-time data systems*. Manning.

McGregor A. (2014). Graph stream algorithms: a survey. *ACM SIGMOD Record*, vol. 43, n° 1, p. 9–20.

Meehan J., Zdonik S., Tian S., Tian Y., Tatbul N., Dziedzic A. *et al*. (2016). Integrating real-time and batch processing in a polystore. In *High performance extreme computing conference (hpec), 2016 ieee*, p. 1–7.

Mishne G., Dalton J., Li Z., Sharma A., Lin J. (2013). Fast data in the era of big data: Twitter's real-time related query suggestion architecture. In *Proceedings of the 2013 acm sigmod international conference on management of data*, p. 1147–1158.

Munshi A. A., Mohamed Y. A.-R. I. (2018). Data lake lambda architecture for smart grids big data analytics. *IEEE Access*, vol. 6, p. 40463–40471.

Noghabi S. A., Paramasivam K., Pan Y., Ramesh N., Bringhurst J., Gupta I. *et al*. (2017). Samza: stateful scalable stream processing at linkedin. *VLDB Endowment*, vol. 10, n° 12, p. 1634–1645.

Pal G., Li G., Atkinson K. (2018). Multi-agent big-data lambda architecture model for e-commerce analytics. *Data*, vol. 3, n° 4, p. 58.

Silva J. A., Faria E. R., Barros R. C., Hruschka E. R., De Carvalho A. C., Gama J. (2013). Data stream clustering: A survey. *ACM Computing Surveys (CSUR)*, vol. 46, n° 1, p. 13.

Singh D., Reddy C. K. (2015). A survey on platforms for big data analytics. *Journal of big data*, vol. 2, n° 1, p. 8.

Yang F., Merlino G., Ray N., Léauté X., Gupta H., Tschetter E. (2017). The RADStack: Open source lambda architecture for interactive analytics. In *Proceedings of the 50th hawaii international conference on system sciences.*

Méthode basée sur les patterns pour la détection simultanée d'anomalies multiples dans les réseaux de capteurs

Ines Ben Kraiem [1], **Faiza Ghozzi** [2], **Andre Peninou** [1], **Olivier Teste** [1]

1. *Université de Toulouse, UT2J, IRIT, Toulouse, France*
   {ines.ben-kraiem,andre.peninou,olivier.teste}@irit.fr
2. *Université de Sfax, ISIMS, MIRACL, Sfax, Tunisia*
   faiza.ghozzi@isims.usf.tn

RÉSUMÉ. *La détection d'anomalies dans les applications réelles de distribution de fluide est une tâche difficile, en particulier lorsque l'on cherche à détecter simultanément différents types d'anomalies. La résolution de ce problème est importante dans plusieurs domaines par exemple, dans les applications de gestion et de supervision de bâtiments. Dans cet article, nous présentons l'algorithme CoRP "Composition of Remarkable Points", une approche configurable basée sur la modélisation de patterns de détection simultanée d'anomalies multiples. CoRP applique un ensemble de patterns, défini par l'utilisateur, afin d'annoter (labels) les points remarquables dans une série temporelle uni-variée, puis détecte les anomalies par composition de labels. En comparant avec des algorithmes de la littérature, notre approche se montre plus robuste et plus précise pour détecter tous les types d'anomalies observées dans des déploiements réels. Nos expérimentations reposent sur des données du monde réel et des données de benchmark issues de la littérature.*

ABSTRACT. *The detection of anomalies in real fluid distribution applications is a difficult task, especially, when we seek to simultaneously detect different types of anomalies and possible sensor failures. Resolving this problem is increasingly important in many areas, for example, in building management and supervision applications. In this paper we introduce CoRP "Composition of Remarkable Points" a configurable approach based on pattern modelling, for the simultaneous detection of multiple anomalies. CoRP evaluates a set of patterns that are defined by users, in order to tag the remarkable points (labels) in a univariate time series and then detects among them the anomalies by composition of labels. By comparing with literature algorithms, our approach appears more robust and accurate to detect all types of anomalies observed in real deployments. Our experiments are based on real-world data and benchmark data from the literature.*

MOTS-CLÉS : *Réseaux de capteurs, Détection d'anomalies, Méthode basée sur les patterns.*

KEYWORDS: *Sensor networks, Anomaly detection, Pattern-based method.*

1. Introduction

Les réseaux de capteurs jouent un rôle important dans la supervision et l'exploration des réseaux de distribution de fluides (e.g., énergie, eau, chauffage) à l'échelle d'un campus et plus largement d'une ville ou d'une région. L'exploitation de ces réseaux repose sur des données relevées par des capteurs. Ces données comportent des anomalies qui nuisent à la supervision (e.g., fausses alarmes, arrêts). Par exemple, la figure 1 illustre un changement brusque (représenté par une croix) dans les mesures de capteurs, engendrant un changement de niveau permanent suite à un problème de matériel (e.g., capteurs endommagés, changement de capteur). Les triangles illustrent plusieurs pics représentant des défauts de lecture liés à un événement imprévu (e.g., panne, rupture). Enfin, le rectangle représente un décalage constant dans les mesures (dû à un problème de communication). Dans un tel cas de figure, les mesures du capteur peuvent différer de leurs valeurs réelles ou attendues et se transformer ainsi en anomalies ce qui rend la tâche d'exploitation plus difficile et complexe. C'est dans cette optique, que la détection d'anomalies apparaît comme étant le moyen pour identifier les évènements anormaux et détecter des comportements qui ne sont pas conformes au comportement attendu (Chandola *et al*., 2009). Au-delà de la supervision des réseaux de capteurs, il existe un large éventail d'applications pour lesquelles il est primordial de détecter les anomalies pour faciliter l'analyse des données notamment la détection d'intrusions, la détection de dommages industriels, la détection d'anomalies dans l'imagerie médicale, la détection d'anomalies dans les données textuelles, la surveillance, la détection de fraude dans les transactions financières, etc (Hodge, Austin, 2004). Plusieurs techniques ont été proposées dans la littérature et classées selon les domaines d'applications ou les types d'anomalies à détecter (Chandola *et al*., 2009). Néanmoins, ces techniques ne permettent pas toujours de détecter tous les types d'anomalies, obligeant les applications à utiliser plusieurs méthodes pour détecter de multiples anomalies de nature diverse.

Cet article se place dans le cadre d'applications réelles ayant des anomalies spécifiques au métier (gestion des fluides sur le campus de Rangueil-Toulouse). La problématique est de trouver une méthode permettant de détecter de multiples anomalies de différents types observées lors de déploiements réels tout en maximisant le nombre d'anomalies détectées et en minimisant les erreurs. Nous traitons dans ce contexte des séries temporelles uni-variées. La difficulté de disposer d'une technique robuste pour détecter l'ensemble des anomalies nous amène à définir une nouvelle méthode configurable nommée **CoRP** "Composition of Remarkable Points". Cette méthode permet, premièrement, de détecter des points qui paraissent remarquables dans les séries temporelles en évaluant des patterns et, deuxièmement, de créer des compositions de points remarquables utilisées pour identifier de multiples anomalies.

La suite de cet article est structurée comme suit. Dans la section 2, nous étudions quelques techniques et algorithmes de la littérature traitant la détection d'anomalies dans les séries temporelles. Dans la section 3, nous décrivons et détaillons notre approche. La section 4 présente les expérimentations effectuées, d'une part, sur notre

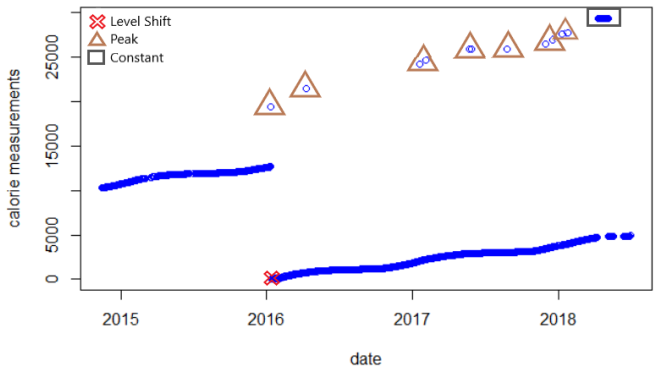


Figure 1. Exemple d'anomalies dans les mesures de capteurs.

étude de cas réel et, d'autre part, sur des données de référence. Enfin, nous concluons avec les perspectives et les recherches ultérieures possibles pour notre travail.

2. État de l'art sur la détection d'anomalies

La plupart des recherches existantes en détection d'anomalies portent sur divers domaines d'applications (Chandola *et al*., 2009) (Hodge, Austin, 2004) (Sreevidya *et al*., 2014). Ces articles identifient de nombreuses techniques de détection d'anomalies en fonction du domaine d'application. Les principales approches sont basées sur le regroupement, la classification, les statistiques, les voisins les plus proches, la régression, la décomposition spectrale et la théorie de l'information.

Sharma *et al*. (2010) ont exploré des techniques pour détecter des anomalies courtes, de bruits et constantes. Pour cela, quatre classes de méthodes de détection d'anomalies sont proposées : méthodes basées sur des règles, méthodes d'estimation des moindres carrés, méthodes par apprentissage (HMM) et méthodes basées sur l'analyse des séries temporelles (ARIMA). Bien que chacune de ces méthodes détecte les types d'anomalies spécifiques, elles génèrent toujours des erreurs, en particulier dans un contexte d'anomalies multiples. Pour cette raison, des méthodes hybrides ont été proposées, Hybrid (U) et Hybrid (I), afin de réduire le nombre de faux négatifs et de faux positifs en combinant les résultats de chaque méthode (Sharma *et al*., 2010).

Yao *et al*. (2010) proposent un algorithme appelé analyse de séquence segmentée (SSA) qui consiste à comparer les mesures à évaluer avec une série temporelle de référence. Cette méthode ne permet pas de détecter avec précision toutes les anomalies. Pour cette raison, les auteurs ont proposé une approche hybride qui est une combinaison de SSA et de la méthode basée sur les règles. Ainsi, leur approche consiste à appliquer la méthode basée sur les règles dans une première phase afin de détecter les anomalies courtes, puis l'algorithme SSA pour détecter les anomalies restantes.

Il existe d'autres méthodes de détection d'anomalies sur des données unies-variées telles que le test ESD généralisé (Extreme Studentized Deviate) (Rosner, 1983) et Change Point (Basseville *et al*., 1993) (Aminikhanghahi, Cook, 2017). L'algorithme ESD utilise des fonctions statistiques telles que la moyenne et la déviation standard pour la détection d'anomalies. ESD nécessite de spécifier une limite supérieure pour le nombre probable d'anomalies existantes ; ceci n'est pas possible pour toutes les applications. La méthode de changement de point (Change Point) détecte les changements de distribution (e.g., moyenne, variance, covariance) dans les mesures du capteur (Basseville *et al*., 1993). Cette méthode détecte chaque changement sous forme d'anomalies.

D'autres méthodes sont basées sur la technique du voisin le plus proche pour la détection d'anomalies et peuvent être regroupées en deux catégories (Chandola *et al*., 2009) : (1) les techniques qui utilisent la distance d'une instance de données à son kème voisin le plus proche comme score d'anomalie (Upadhyaya, Singh, 2012).
(2) les techniques qui calculent la densité relative de chaque instance de données pour calculer son score d'anomalies, par exemple, l'algorithme LOF (Local Outlier Factor) (Breunig *et al*., 2000).

Bien que chacune de ces méthodes ait été conçues pour la détection d'anomalies, nous pensons qu'elles ne satisfont pas toutes les propriétés souhaitables, y compris la détection de multiples types d'anomalies observées de manière simultanée dans les déploiements réels avec un taux d'erreurs faible. De plus, plusieurs méthodes parmi les méthodes mentionnées nécessitent un prétraitement ou un post-traitement à travers des méthodes hybrides pour améliorer leurs résultats. Pour évaluer les performances de ces méthodes, nous avons sélectionné des algorithmes appartenant à différentes techniques et proches de la détection des types d'anomalies recherchées. Nous effectuons une comparaison entre ces méthodes appliquées sur notre étude de cas dans la section 4.

Exploration des méthodes de détection existantes. Dans notre étude, nous avons exploré cinq méthodes appartenant à quatre techniques différentes pour détecter les types d'anomalies observées dans notre application.

– Méthode basée sur les règles : nous avons utilisé deux règles pour détecter les anomalies courtes (changement anormal) et les anomalies constantes (pas de variation) (Sharma *et al*., 2010). *La règle d'anomalie courte* traite la série temporelle en comparant à chaque fois deux observations successives : on détecte une anomalie si la différence entre ces observations est supérieure à un seuil donné. Pour déterminer automatiquement le seuil de détection, nous avons utilisé l'approche basée sur l'histogramme (Ramanathan *et al*., 2006). *La règle d'anomalie constante* calcule l'écart-type pour un ensemble d'observations successives. Si cette valeur est égale à zéro l'ensemble est déclaré comme une anomalie.

– Méthode basée sur la densité : cette approche consiste à comparer la densité autour d'un point par rapport à la densité de ses voisins locaux. Breunig *et al*. (2000) ont proposé l'algorithme LOF. Dans cette méthode, les scores des anomalies sont mesurés en utilisant un facteur de valeur aberrante local, qui est le rapport entre la

densité locale autour de ce point et la densité locale autour de ses plus proches voisins. Le point dont la valeur LOF est élevée est déclaré une anomalie.

– Méthode basée sur les statistiques : premièrement, nous avons utilisé la méthode ESD pour la détection automatique des anomalies locales et globales. Deuxièmement, nous avons utilisé la méthode Change Point pour détecter le changement de niveau.

– Méthode basée sur l'analyse des séries temporelles : le principe de cette approche est d'utiliser les corrélations temporelles pour modéliser et prédire les valeurs de la série temporelle. Nous avons utilisé le modèle ARIMA (AutoRegressive Intergrated Moving Average) pour la création du modèle de prédiction selon l'approche décrite par Chen et Liu (1993). Une mesure de capteur est comparée à sa valeur prédite pour déterminer si elle est une anomalie.

Il existe des implémentations open source pour des algorithmes tels que LOF, ARIMA, S-H-ESD et Change Point, ((Hochenbaum *et al*., 2017) (Cleveland *et al*., 1990) (Rosner, 1983) (Aminikhanghahi, Cook, 2017)) que nous avons utilisés pour les expérimentations. En revanche, nous avons mis en œuvre d'autres approches (règle courte et règle constante) en fonction des sources disponibles.

Le tableau 1 représente la synthèse des méthodes que nous avons explorées pour détecter chaque type d'anomalies détectées lors de déploiements réels et présentées dans la figure 1.

Tableau 1. Les méthodes de détection d'anomalies étudiées.

Type d'anomalies	Méthodes de détection
Changement anormal	Règle courte, $ARIMA, LOF, S-H-ESD$
Valeurs constantes	Règle constante
Changement de niveau	$ARIMA$, Change Point

3. Méthodologie CoRP pour la détection simultanée d'anomalies multiples

En situation d'exploitation réelle, la supervision des réseaux de capteurs se fait par les experts (ingénieurs d'exploitation, techniciens de maintenance, etc) en observant les courbes afin de détecter les points qui semblent remarquables et qui montrent des comportements inhabituels. Typiquement, ce sont les mesures des capteurs de notre étude de cas illustrées figure 1. Ces points remarquables sont les variations inhabituelles entre les points successifs d'une série temporelle et qui sont les marqueurs (ou indices) de possibles anomalies. Afin de mettre en œuvre cette démarche de détection de façon automatique, nous avons créé notre approche configurable, appelée CoRP, de détection d'anomalies. Elle est basée sur des patterns pour la détection des points remarquables et sur des compositions de ces points afin d'identifier les anomalies.

3.1. Notations utilisées

Définition 1 *Une série temporelle* est composée d'observations ou de points successifs collectés séquentiellement dans le temps à intervalle régulier. Ces points repré-

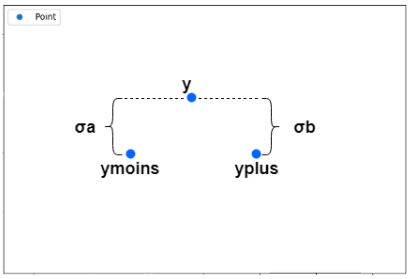


FIGURE 2. *Labellisation d'un point remarquable "y" par un pattern σ_a et σ_b.*

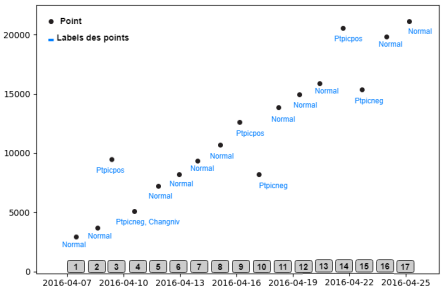


FIGURE 3. *Labellisation d'une série de points remarquables (algorithme 1).*

sentent les mesures associées à un horodatage indiquant l'heure de sa collecte. Soit $Y_i = \{y_1, y_2, y_3....\}$ une série temporelle représentant la séquence des points ou mesures de capteurs collectées pour chaque point i $\in \mathbb{N}$.

Définition 2 *Un point (ou mesure)* est composé d'une valeur et d'un horodatage. Dans cet article, on note un point $y_i = (t_i, v_i)$ tel que t_i est l'horodatage de y_i (noté $t(y_j)$) et v_i est la valeur de y_i (noté $v(y_j)$).

3.2. Description de CoRP

En s'appuyant sur l'expérience des experts (détection des points remarquables puis identification des anomalies), l'algorithme CoRP est construit en deux phases. La première consiste à détecter et annoter les points considérés comme remarquables dans la série temporelle. La deuxième phase consiste à créer une composition de points remarquables permettant d'identifier les anomalies.

3.2.1. Détection des points remarquables

La détection des points remarquables est réalisée à partir des patterns de détection.

Définition 3 *Un pattern p* est défini par un triplet $p = (l, \sigma_a, \sigma_b)$ où l est un label qui étiquettera un point détecté comme remarquable par le pattern, σ_a et σ_b sont deux seuils utilisés pour décider si un point donné est remarquable. Un pattern est appliqué sur trois points consécutifs y_{j-1}, y_j, y_{j+1} d'une série temporelle Y, points qui sont notés y_{moins}, y, y_{plus}. σ_a correspond à l'écart entre $v(y_{moins})$ et $v(y)$ tandis que σ_b correspond à l'écart entre $v(y)$ et $v(y_{plus})$ comme illustré figure 2. Lorsqu'un pattern est vérifié sur y_{moins}, y, y_{plus}, le label du pattern est utilisé pour étiqueter le point y.

Définition 4 *Une série temporelle labellisée* est une série temporelle de points sur lesquels sont ajoutés les labels détectés par les patterns.

Définition 5 *Un point remarquable* y_i d'une série temporelle labellisée est défini par un triplet (t_i,v_i, L_i) où t_i est son horodatage, v_i est sa valeur et $L_i = \{l_1, l_2, ...\}$ est une liste de labels caractérisant le point.

Les patterns sont donc utilisés pour détecter les points remarquables et leurs ajouter le label correspondant. Ainsi, la liste des labels d'un point remarquable est constituée des labels des différents patterns vérifiés sur ce point. La figure 3 illustre un extrait d'une série temporelle labellisée des données d'index qui ont tendance à croître. Ainsi, la figure 3 comporte 4 exemples de labels (Normal, Ptpicpos, Ptpicneg, Changniv). Notons le point 4 qui comportent 2 labels.

Exemple. Des exemples de patterns utilisés afin de labelliser la courbe de la figure 3 sont : (i) un "Point Pic Positif" remarquable (Ptpicpos, 100, 100) où Ptpicpos représente le label descriptif du pattern, σ_a=100 et σ_b=100 ; (ii) un "Point Pic Négatif" remarquable (Ptpicneg, -100, -100) ; et (iii) un "Changement de Niveau" remarquable (Changniv, -1000, -100).

L'algorithme 1, appelé EvaluatePattern, permet d'évaluer un pattern à l'aide de règles. Cette fonction prend en entrée trois points successifs notés y_{moins}, y et y_{plus} et le pattern à évaluer, et renvoie le résultat de l'évaluation. Différentes règles de vérifications sont appliquées en fonction des signes de σ_a et σ_b. Les règles pour comparer y_{moins} et y en fonction de σ_a sont les suivantes : (i) Si $\sigma_a > 0$, la règle à vérifier est $v(y) >= v(y_{moins}) + \sigma_a$; (ii) Si $\sigma_a < 0$, la règle à vérifier est $v(y) <= v(y_{moins}) + \sigma_a$; et (iii) Si $\sigma_a = 0$, la règle à vérifier est $v(y) = v(y_{moins})$. Les règles pour comparer y et y_{plus} en fonction de σ_b sont similaires (voir l'Algorithme1).

Algorithm 1 Evaluation d'un pattern

function BOOLEAN EVALUATEPATTERN($y_{moins}, y, y_{plus}, p$)
Input $y_{moins}, y, y_{plus}, p = (l_p, \sigma_a, \sigma_b)$
Output Boolean
  if $p.\sigma_a > 0$ **then** leftRemarkable<-($v(y) \geq v(y_{moins}) + p.\sigma_a$? true : false)
  else if $p.\sigma_a < 0$ **then** leftRemarkable<-($v(y) \leq v(y_{moins}) + p.\sigma_a$? true : false)
  else if $p.\sigma_a = 0$ **then** leftRemarkable<-($v(y) = v(y_{moins})$? true : false)
  end if
  if $p.\sigma_b > 0$ **then** rightRemarkable<-($v(y) \geq v(y_{plus}) + p.\sigma_b$? true : false)
  else if $p.\sigma_b < 0$ **then** rightRemarkable<-($v(y) \leq v(y_{plus}) + p.\sigma_b$? true : false)
  else if $p.\sigma_b = 0$ **then** rightRemarkable<-($v(y) = v(y_{plus})$? true : false)
  end if
return (leftRemarkable and rightRemarkable)
end function

L'algorithme 2 fait appel à la fonction EvaluatePattern pour traiter une série temporelle. Il prend en entrée la série temporelle initiale et la liste des patterns et renvoie une nouvelle série temporelle labellisée. Le traitement consiste à parcourir la série temporelle et la liste des patterns. Pour chaque point et pour chaque pattern p, la fonction EvaluatePattern est appelée afin d'ajouter (ou pas) le label de p au point évalué.

Algorithm 2 Detection de points remarquables

Input $Y = \{y_1, y_2, y_3....\}$, $P = \{p_1, p_2, p_3....\}$
Output Y_L série temporelle labellisée

for i in range(2..|Y|-1) **do**
  for k in range(1..|P|) **do**
    if EvaluatePattern(y_{i-1},y_i,y_{i+1},p_k) **then**
      L_i <- $L_i + p_k.l$
    end if
  end for
end for
  return Y_L

3.2.2. Composition de patterns

Le résultat de l'algorithme 2 est une série temporelle labellisée à l'aide de patterns. Comme montré dans la figure 4, un point peut être labellisé par un ou plusieurs patterns. A partir d'un sous-ensemble de points d'une série temporelle labellisée, on construit par concaténation des labels L_i de ces points remarquables, une chaîne de labels. Cette chaîne de labels est utilisée pour détecter une anomalie. Une anomalie est reconnue par une composition de labels et par la vérification de conditions sur les valeurs des points.

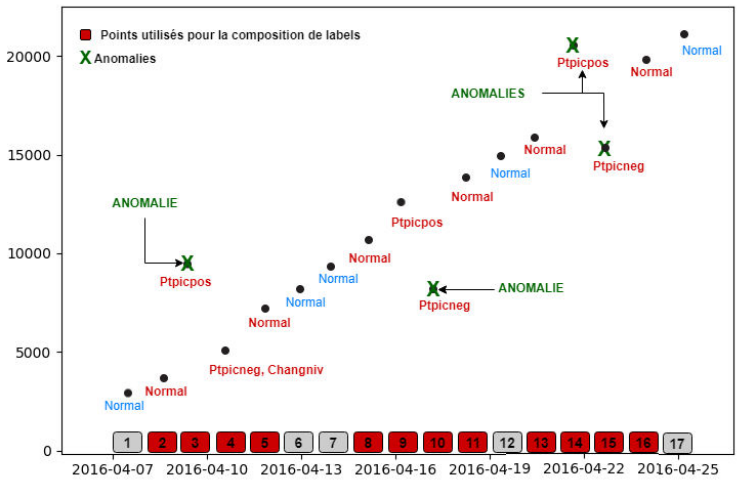


Figure 4. Résultat de la phase 2 de l'algorithme CoRP.

Définition 6 *Une anomalie* est un ou plusieurs points remarquables appartenant à un sous-ensemble de points pour lequel sont vérifiées, d'une part, une composition des labels de ces points et, d'autre part, une condition exprimée sur les valeurs de ces points. L'anomalie est enfin identifiée sur un ou plusieurs points de cette composition.

Pour définir une composition de labels, nous proposons une grammaire, illustrée dans la figure 5, qui définit les éléments d'une composition de labels. La grammaire permet de définir les labels possibles (un ou plusieurs) sur des points successifs permettant de reconnaître une composition de labels. La grammaire part des labels posés sur les points (<label>). Les labels peuvent être combinés sur un seul point avec des expressions logiques AND, OR et NOT (<label-comp> and <point-label>). Par exemple "l1 AND NOT l2 AND l3" désigne un point labellisé avec l1, labellisé avec l3 et non labellisé l2. Chaque composition de labels sur un point unique peut être répétée sur des points successifs par des quantificateurs : ?, + et * (<label-enum>). Par exemple "(l1) +" signifie que la composition doit comporter un ou plusieurs points successifs labellisés l1. La composition de labels finale est créée à travers une succession de labels séparés par "." (< composition >). Par exemple, "l1. (l2) *. l1 OR l3" signifie un point labellisé par l1 suivi de zéro ou plus de points labellisés par l2 suivi d'un point labellisés par l1 ou par l3.

```
<composition> ::= <label-enum> ("." <label-enum>) *

<label-enum> ::= <label-comp>
           |   "(" <label-comp> ")" "?"
           |   "(" <label-comp> ")" "*"
           |   "(" <label-comp> ")" "+"

<label-comp> ::= <point-label> ("OR" <point-label>) *
           |     <point-label> ("AND" <point-label>) *
           |     <point-label>

<point-label> ::= <label>
           |      "NOT" <label>

<label> ::= list of labels defined by patterns
```

Figure 5. Grammaire pour la définition d'une composition de labels.

Définition 7 *Une composition de labels* permettant de reconnaître une anomalie est composée de trois parties :

– *composition* : la composition des labels des points remarquables qui est une séquence de points comportant des labels définies selon la grammaire présentée figure 5 ;

– *condition* : une condition entre les valeurs des points reconnus (ceux correspondants à la séquence des labels). Une même composition de labels peut correspondre à différentes anomalies. Cette condition est créée à l'aide des opérateurs (<, <=, etc) permettant de comparer des valeurs et des opérateurs logiques (AND/OR/NOT) permettant de combiner des comparaisons. Afin d'éviter l'utilisation de la notation $v(y_i)$, nous notons par v_i la valeur du ième point reconnu par la composition, v_1 le premier et v_n le dernier ; notons que le nombre de points impliqués dans la composition peut être variable compte tenu des quantificateurs utilisables dans la composition ;

– *conclusion* : l'anomalie identifiée pour laquelle sont précisés son type (nom de l'anomalie) et la liste des valeurs (points) où se situe l'anomalie détectée.

A titre d'exemple, nous donnons les compositions de labels pour identifier trois types d'anomalies : (i) anomalie de valeurs constantes, (ii) anomalie de valeurs en pic négatif, et (iii) anomalie de valeurs en pic positif ; cette dernière est possiblement reconnue à partir de 2 compositions de labels.

Label-composition 1
composition : Normal . Ptpicpos . Ptpicneg . Normal
condition : $v_2 > v_4$ and $v_3 > v_1$
conclusion : positive peak -> v_2
Label-composition 2
composition : Normal . Ptpicpos . Ptpicneg . Normal
condition : $v_2 < v_4$ and $v_3 < v_1$
conclusion : negative peak -> v_3
Label-composition 3
composition : Begincstpos . Cst* . Endcstneg
condition : $v_1 == v_2$ and $v_{n-1} == v_n$
conclusion : constant -> all
Label-composition 4
composition : Normal . Ptpicpos . Ptpicneg AND Changnivneg . Normal
condition : $v_2 > v_4$ and $v_3 > v_1$
conclusion : positive peak -> v_2

Exemple. Considérons les sous-ensembles de points présentés sur la figure 4 en rouge. Les points d'indices 2 à 5 donnent la série de labels suivante : (Normal . Ptpicpos . Ptpicneg and Changniv . Normal) détectée par la composition de labels 4 de l'exemple. Cette composition permet donc de détecter l'anomalie de pic positif en 3. Les points d'indices 8 à 11, déclenchent les compositions de labels 1 et 2. Pour Label-composition 1, la condition est $v_9 > v_{11}$ et $v_{10} > v_8$ qui est fausse donc la composition n'est pas valide. Pour Label-composition 2, la condition est $v_9 < v_{11}$ et $v_{10} < v_8$ qui est vraie donc la composition est valide et l'anomalie Pic Négatif est reconnue en v_{10}.

Enfin, nous avons mis en œuvre un algorithme capable de parcourir une liste labellisée Y_L et vérifier, à partir de chaque point, quelles compositions de labels s'appliquent pour identifier les anomalies (et les points correspondants).

4. Expérimentations

4.1. Description d'étude de cas

Le domaine d'application traité dans cet article est le réseau de capteurs du Service de gestion et d'exploitation (SGE) du campus de Rangueil rattaché au rectorat de Toulouse. Ce service exploite et entretient le réseau de distribution à partir des données liées aux différentes installations. Plus de 600 capteurs de différents types de fluides (calories, eau, air comprimé, électricité et gaz), disséminés dans plusieurs bâtiments, sont gérés par les systèmes de supervision SGE. Pour cet article, nous nous sommes concentrés sur les données de calories et nous avons traité 25 capteurs. Ainsi, nous

analysons les mesures de calories collectées chaque jour pendant plus de trois ans par 25 capteurs déployés dans différents bâtiments (1453 données par capteur soit 36325 points de données au total). Les mesures de ces capteurs sont rassemblées à une fréquence régulière et représentent les *index* (lectures de capteurs) et sont ensuite utilisées pour mesurer les *quantités d'énergie consommées* (par différences de valeur d'index successives). Nous avons pu identifier les types d'anomalies et les points concernés présents dans les données de capteurs de calorie grâce aux connaissances acquises par les experts du SGE et à travers une inspection manuelle d'un ensemble de capteurs de même type que les capteurs étudiés.

Les défauts présentés figure 1 sont extraits de ces mêmes ensembles de données. L'anomalie prédominante est constituée par les valeurs constantes, près de 8578 observations sur toutes les données, suite à un arrêt de capteurs. Nous avons également trouvé parmi ces valeurs plusieurs constantes avec un décalage. Généralement, une constante avec un décalage de niveau commence par un pic positif ou négatif. Ensuite, il existe beaucoup de changements anormaux tels que des pics positifs ou négatifs, près de 380. Enfin, il y a huit changements de niveau dus au changement de capteur. Afin de détecter les points remarquables et les anomalies, nous avons construit, avec l'aide des experts, 14 patterns et 12 compositions de labels pour détecter les anomalies décrites ci-dessus.

4.2. Expérimentation sur des données réelles (séries croissantes et séries variables)

Dans cette partie, nous explorons les différentes méthodes de détection d'anomalies décrites dans le tableau 1. Notre motivation à envisager ces méthodes est d'élargir le champ d'analyse afin de tester leur efficacité dans la détection d'anomalies cherchées dans cet article et de comparer les résultats avec notre approche. Comme indiqué ci-dessus, ces techniques se sont révélées efficaces pour détecter les anomalies dans les données du capteur. Cependant, comme nous le verrons à travers les résultats des expériences, aucune de ces méthodes n'est efficace pour détecter tous les types d'anomalies de manière simultanée dans les déploiements réels. Nous présentons une évaluation des méthodes suivantes : Règle Courte (noté SR), Règle Constante (noté CR), LOF, ARIMA, S-H-ESD et Change Point (noté LS). Nous avons appliqué ces méthodes par catégorie d'anomalies comme indiqué dans le tableau 1. Afin d'évaluer leurs performances, nous utilisons le nombre de vrais positifs (vraies anomalies détectées), le nombre de faux positifs (fausses anomalies détectées) et le nombre de faux négatifs (vraies anomalies non détectées) en tant que métriques d'évaluation.

Nous présentons les résultats de LOF dans un graphique séparé pour plus de lisibilité (figure 6B). Ces méthodes ne sont pas entièrement automatisées et nous devons donc sélectionner les paramètres tels que le seuil pour la Règle Courte, le nombre de voisins et le seuil pour évaluer le score du degré d'anomalie pour LOF ou le type de modèle pour ARIMA, etc. Pour LOF, nous avons fait varier le choix de K (nombre de voisins) dans une plage de 30 à 10 afin d'évaluer l'influence de ce paramètre sur le résultat de la détection (cf. haut de la figure 6B) et nous avons déterminé un seuil

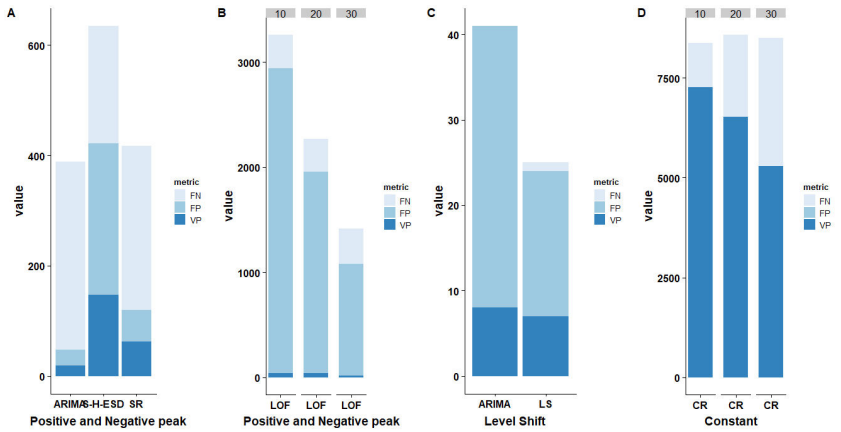


Figure 6. Evaluation des méthodes de détection d'anomalies sur les données d'index.

= 1,5 qui correspond à une distribution standard. Pour la Règle Courte, nous avons choisi un seuil en utilisant la méthode basée sur l'histogramme décrite dans la section 2. Enfin, pour la règle constante, nous avons fait varier le choix de la taille de la fenêtre coulissante dans une plage de 30 à 10 comme le montre la figure 6D (haut de la figure).

En se basant sur les résultats présentés figures 6A, 6B, 6C, et 6D, nous pouvons faire les observations suivantes : LOF est la méthode qui génère le plus de faux positifs tandis ARIMA génère le plus de faux négatifs. La méthode S-H-ESD est celle qui permet de détecter le plus de vrais positifs par rapport à LOF, ARIMA et RC, par contre elle génère beaucoup de faux positifs et de faux négatifs. La Règle Courte détecte moins d'anomalies en comparaison de S-H-ESD. Cependant, il en résulte moins de faux positifs que les autres méthodes. Également, nous pouvons dire que le nombre de voisins égal à 20 est le choix le plus approprié pour détecter le plus grand nombre d'anomalies pour l'algorithme LOF. Cependant, pour la Règle de Constante, il est important d'utiliser une taille de fenêtre suffisamment petite pour gérer des données contenant plusieurs anomalies constantes (ici 10).

Le tableau 2 présente les résultats de ces méthodes en fonction de la précision, du rappel et de la F-mesure sur les données d'index (série croissante) et sur les données de consommation (série variable).

Concernant les données d'index, en se basant sur ce tableau et la figure 6 nous déduisons que : (i) l'efficacité de la Règle Constante ou de la méthode LOF dépend fortement du choix de la fenêtre glissante ou du nombre de voisins ; (ii) la méthode Change Point fonctionne bien lorsqu'il y a réellement un changement de niveau dans la série temporelle, mais cependant, en cas d'absence d'anomalie, sa précision est faible ; et (iii) entre la Règle Courte, ARIMA et S-H-ESD, la Règle Courte est la plus précise et ARIMA est la moins efficace pour détecter un changement anormal. En comparant avec ces méthodes, notre algorithme CoRP arrive à détecter de multiples

Tableau 2. Comparaison des méthodes de détection d'anomalies sur les données d'index et de consommation.

Données	Séries croissantes (Index)			Séries variables (Consommation)		
	Pre	Rec	$F-m$	Pre	Rec	$F-m$
SR	0.52	0.17	0.32	0.66	0.63	0.64
CR	1	0.80	0.88	1	0.72	0.83
LOF	0.022	0.12	0.022	0.39	0.78	0.52
$S-H-ESD$	0.34	0.40	0.36	0.41	0.80	0.54
$ARIMA$	0.30	0.07	0.11	0.66	0.25	0.36
LS	0.29	0.87	0.43	X	X	X
$CoRP$	1	1	1	1	0.98	0.98

types d'anomalies simultanément avec une meilleure précision et un meilleur rappel. En effet, CoRP fonctionne très bien sur les données d'index représentant notre étude de cas.

Pour évaluer davantage notre algorithme et le confronter aux méthodes de détection d'anomalies, nous avons utilisé les données de consommation du SGE. Nous avons donc pris les mesures provenant des 25 capteurs. Les données de consommation sont des données saisonnières et leur évolution quotidienne, contrairement aux données d'index, est variable. Nous avons inspecté manuellement un ensemble de données de même type pour comprendre leurs variations et créer les patterns de détection des points remarquables qui peuvent exister puis les compositions de labels pour détecter les anomalies. Les anomalies observées dans ces données sont les suivantes : pics positifs et négatifs, anomalies constantes, anomalies constantes commençant et se terminant par un décalage important. Ainsi, toujours avec l'aide des experts, nous avons créé 9 patterns afin de détecter les points remarquables et 5 compositions de labels pour détecter les anomalies.

Nous rapportons les résultats des algorithmes sur les données de consommation dans le tableau 2 (séries variables). Comme les données ne sont pas stationnaires, nous n'avons pas appliqué l'algorithme Change Point parce qu'il n'existe pas de changement de niveau dans ces données à détecter. Ce tableau montre que les algorithmes de la littérature sont beaucoup plus efficaces sur les données de consommation en comparant avec les résultats sur les données d'index. Mais même sur ce type de données, notre approche a obtenu le meilleur résultat de F-mesure en comparant avec les autres algorithmes. En effet, CoRP a détecté le plus d'anomalies avec le moins d'erreurs possibles, avec une précision égale à 1 et un rappel égal à 0,98. Notons que, les résultats de la méthode à base de règles (SR, CR) et de la méthode ARIMA ont une meilleure précision par rapport à LOF et SH-ESD. Enfin, SH-ESD est la méthode la plus proche du meilleur résultat en termes de rappel avec une valeur égale à 0,80. Toutefois, il faut noter que ces algorithmes n'arrivent pas détecter simultanément tous les types d'anomalies observées dans les déploiements réels, ce qui signifie que chaque algorithme est efficace dans un type spécifique. La particularité de notre méthode est que

nous pouvons définir les patterns en fonction de nos besoins afin de détecter avec une grande précision et efficacité de multiples anomalies simultanées.

4.3. Expérimentation sur des données de la littérature (séries variables)

Afin d'évaluer l'algorithme dans un autre contexte, nous avons utilisé les ensembles de données proposés dans le package d'implémentation de la méthode ARIMA (Lacalle, 2016). Parmi ces données, nous avons exploré les données de HIPC (Harmonised Indices of Consumer Prices). Ces ensembles de données représentent les indices harmonisés des prix à la consommation dans la zone euro. Nous avons également exploré les données IPI (Industrial Production Indices). Ces données représentent les indices de la production industrielle dans le secteur manufacturier des pays de l'Union monétaire européenne (Lacalle, 2016). Chacun de ces ensembles de données contient plusieurs séries temporelles qui présentent des données mensuelles de 1995 à 2013. Un premier sous-ensemble comportant deux séries temporelles de ces deux ensembles de données a permis de mener les expérimentations. Chacune de ces séries contient 229 mesures avec 5 anomalies en HIPC et 4 anomalies en IPI. Ces anomalies sont variées : AO (Additive Outlier), TC (Temporary Changes) ou LS (Level Shift). Un deuxième sous-ensemble de séries nous a permis de spécifier les patterns en définissant un pattern différent par type d'anomalies pour labelliser les points remarquables dans la série temporelle (3 patterns). Ensuite, nous avons procédé à une composition de ces labels pour détecter les anomalies (4 compostions de labels).

Tableau 3. Comparaison de méthodes de détection d'anomalies sur des données de benchmark.

Datasets	HIPC		IPI	
Evaluation / Algorithm	Precision	Recall	Precision	Recall
CoRP	1	0.80	0.75	0.75
ARIMA	1	1	1	1
LOF	0.11	0.20	0	0
S-H-ESD	0.20	0.20	0.33	0.25
RC	0	0	0	0
LS	0	0	0	0

Le tableau 3 est une comparaison entre les algorithmes de la littérature et notre algorithme sur les données proposées dans le package ARIMA. Nous n'avons pas testé la Règle Constante dans les ensembles de données HIPC et IPI car les anomalies observées dans ces données ne contiennent pas ce type d'anomalie. Nous avons par conséquent appliqué CoRP, ARIMA, LOF avec un nombre de voisins égal à 20, S-H-ESD, Change Point et la Règle Courte sur ces données. L'algorithme basé sur la Règle Courte et Change Point sont les moins précis parmi ces algorithmes, tandis que notre algorithme est le meilleur parmi eux et peut détecter la majorité des anomalies observées avec peu d'erreurs.

4.4. Temps de calcul

Dans cette partie, nous nous abordons le temps de calcul requis pour les différentes méthodes de la littérature et notre algorithme. Les expériences sont effectuées sur une machine exécutant Windows 10 Professional, un processeur Intel (Core) i5 et 16 Go de RAM. Nous avons utilisé la distribution open source Python 3.7 Anaconda pour développer notre algorithme et R 3.5 pour explorer les algorithmes de la littérature. Nous avons calculé le temps d'exécution sur les 25 séries de données d'index pour chaque algorithme évalué afin de le comparer au temps d'exécution de notre algorithme. Le classement des algorithmes en fonction de leurs performances d'exécution est : la méthode à base de règles et la méthode S-H-ESD sont les plus rapides avec un temps d'exécution de 0.5s. Ensuite, la méthode LOF prend un temps d'exécution égal à 2.5s. Notre algorithme prend une durée d'exécution de 5,40 secondes et enfin ARIMA demande 7,60 secondes.

5. Conclusion

La détection d'anomalies dans les applications de supervision est très importante notamment dans le domaine des réseaux de capteurs. Cet article présente l'approche CoRP basée sur des patterns appliqués aux séries temporelles uni-variées de données de capteurs. Notre méthode est composée de deux étapes : elle marque (labels) tous les points remarquables présents dans la série temporelle sur la base de patterns de détection, puis, elle identifie précisément les multiples anomalies présentes par compositions de labels. Cette approche nécessite l'expertise du domaine d'application pour pouvoir définir efficacement les patterns et les compositions de labels. A l'inverse, les méthodes de la littérature, bien que moins efficaces, ne demandent pas autant d'expertise métier pour être appliquées.

Notre expérimentation est basée sur un contexte réel : les données de capteurs du SGE (service de gestion et d'exploitation du campus de Rangueil à Toulouse). L'évaluation de cette méthode est illustrée en utilisant tout d'abord les données d'index et de consommation des capteurs de calories exploités par le SGE et, en second lieu, en utilisant des jeux de données issus de l'état de l'art. Nous comparons notre algorithme à cinq méthodes appartenant à différentes techniques de détection d'anomalies. Sur la base des critères d'évaluation précision, rappel, f-mesure, nous montrons que notre algorithme est le plus efficace pour détecter de manière simultanée différents types d'anomalies observées lors de déploiements réels en minimisant les fausses détections. Plusieurs extensions à notre approche sont envisagées : (i) afin de moins nécessiter l'intervention d'experts du domaine, utiliser des méthodes d'apprentissage pour construire automatiquement les patterns et/ou les compositions de labels ; (ii) appliquer notre algorithme sur des flux de données réels pour fournir les alarmes en identifiant les anomalies le plus tôt possible.

Remerciements

Ce travail de thèse est financé le service de gestion et d'exploitation (SGE) de Rangueil rattaché au rectorat de Toulouse. Les auteurs remercient le SGE pour avoir fourni l'accès aux données réelles de capteurs. Ils remercient également les experts qui ont aidé à la compréhension de ces données et l'identification des anomalies observées lors de l'exploitation.

Bibliographie

Aminikhanghahi S., Cook D. J. (2017). A survey of methods for time series change point detection. *Knowledge and information systems*, vol. 51, n° 2, p. 339–367.

Basseville M., Nikiforov I. V. *et al*. (1993). *Detection of abrupt changes: theory and application* (vol. 104). Prentice Hall Englewood Cliffs.

Breunig M. M., Kriegel H.-P., Ng R. T., Sander J. (2000). Lof: identifying density-based local outliers. In *ACM SIGMOD record*, vol. 29, p. 93–104.

Chandola V., Banerjee A., Kumar V. (2009). Anomaly detection: A survey. *ACM computing surveys (CSUR)*, vol. 41, n° 3, p. 15.

Chen C., Liu L.-M. (1993). Joint estimation of model parameters and outlier effects in time series. *Journal of the American Statistical Association*, vol. 88, n° 421, p. 284–297.

Cleveland R. B., Cleveland W. S., McRae J. E., Terpenning I. (1990). Stl: A seasonal-trend decomposition. *Journal of Official Statistics*, vol. 6, n° 1, p. 3–73.

Hochenbaum J., Vallis O. S., Kejariwal A. (2017). Automatic anomaly detection in the cloud via statistical learning. *arXiv preprint arXiv:1704.07706*.

Hodge V., Austin J. (2004). A survey of outlier detection methodologies. *Artificial intelligence review*, vol. 22, n° 2, p. 85–126.

Lacalle J. López-de. (2016). tsoutliers R package for detection of outliers in time series. *CRAN, R Package*.

Ramanathan N., Balzano L., Burt M., Estrin D., Harmon T., Harvey C. *et al*. (2006). Rapid deployment with confidence: Calibration and fault detection in environmental sensor networks.

Rosner B. (1983). Percentage points for a generalized esd many-outlier procedure. *Technometrics*, vol. 25, n° 2, p. 165–172.

Sharma A. B., Golubchik L., Govindan R. (2010). Sensor faults: Detection methods and prevalence in real-world datasets. *ACM Transactions on Sensor Networks (TOSN)*, vol. 6, n° 3, p. 23.

Sreevidya S. *et al*. (2014). A survey on outlier detection methods. *International Journal of Computer Science and Information Technologies (IJCSIT)*, vol. 5, n° 6.

Upadhyaya S., Singh K. (2012). Nearest neighbour based outlier detection techniques. *International Journal of Computer Trends and Technology*, vol. 3, n° 2, p. 299–303.

Yao Y., Sharma A., Golubchik L., Govindan R. (2010). Online anomaly detection for sensor systems: A simple and efficient approach. *Performance Evaluation*, vol. 67, n° 11, p. 1059–1075.

Modèle de réseaux multiplexes pour l'étude de l'influence sur Twitter

Lobna Azaza, Éric Leclercq, Marinette Savonnet

Laboratoire d'Informatique de Bourgogne - EA 7534
Univ. Bourgogne Franche-Comté
9, Avenue Alain Savary
F-21078 Dijon - France
prénom.nom@u-bourgogne.fr

RÉSUMÉ. *Les réseaux issus de données réelles tels que Twitter ou Facebook sont appelés réseaux complexes ou graphes de terrain. La plupart des travaux sur ces réseaux ont été abordés sous l'angle de la théorie ou de l'algorithmique des graphes mais très peu sous l'angle de la modélisation des données. Dans ce contexte, nous introduisons les notions fondamentales sur lesquelles est basée notre modélisation des réseaux sociaux en prenant comme exemple Twitter. Notre contribution consiste, dans un premier temps, à déterminer le modèle le plus adapté, c'est-à-dire un réseau multiplexe hétérogène, et à le spécialiser pour spécifier les relations engendrées par les interactions entre les utilisateurs. Dans un second temps, nous exploitons ce réseau multiplexe afin de mesurer l'influence des utilisateurs en incluant les différentes couches (relations). Pour ce faire nous utilisons une extension pour les réseaux multiplexes de l'algorithme PageRank. Notre contribution sur ce point consiste à représenter les données extraites de Twitter dans le modèle de réseau multiplexe que nous avons défini et à exploiter la richesse des relations traduisant l'influence et enfin d'étudier les paramètres qui modifient le comportement de l'algorithme.*

ABSTRACT. *Networks derived from real data such as Twitter and Facebook, etc. are called complex networks or real-world graphs. Most of the work on these networks has been approached from the perspective of graph theory or algorithmic, but very little from the perspective of data modeling. In this context, we introduce the fundamental notions on which our social network modeling is based, using Twitter as an example. Our contribution consists, first, in determining the most appropriate model, i.e., an heterogeneous multiplex network, and specializing it to specify the relationships generated by the interactions between users. In a second step, we operate this multiplex network to measure the influence of users by including the different layers. To do this, we use an extension for the multiplex networks of the PageRank algorithm. Our contribution on this point consists in representing the data extracted from Twitter in the multiplex network model we have defined and exploiting the richness of the relationships that reflect the influence and finally studying the parameters that modify the algorithm's behavior.*

MOTS-CLÉS : *Réseau multiplexe hétérogène, PageRank, Twitter, Influence*

KEYWORDS: *Multiplex Networks, PageRank, Twitter, Influence*

1. Introduction

Dans de nombreux contextes applicatifs (Internet, biologie, Réseaux sociaux), de grands graphes ne montrant pas de structure forte apparente comme des graphes réguliers, des arbres, sont appelés réseaux complexes ou graphes de terrain, par opposition aux graphes explicitement construits par un modèle ou une théorie.

La plupart des travaux sur les réseaux complexes ont été abordés sous l'angle de la théorie ou de l'algorithmique des graphes. En revanche, il existe très peu de travaux de modélisation de tels réseaux et les aspects liés à la sémantique des liens sont peu développés. Notre objectif est de proposer un modèle pour traduire la richesse des données afin de contextualiser les résultats produits par les algorithmes et ainsi aider leur interprétation. Nous prendrons comme exemple de réseau complexe des données réelles issues de *Twitter*. Si le principe de fonctionnement de *Twitter* est extrêmement simple, *tweeter* un texte court, l'utilisation par chacun peut être très variable. *Twitter* présente une grande hétérogénéité sémantique que les modèles de données et les algorithmes doivent prendre en compte. En effet, les liens sont créés par les opérateurs (retweet RT, mentionne @, hashtag #, etc.) qui, selon le domaine et les intentions de l'utilisateur, induisent des sémantiques différentes. L'utilisation de @utilisateur en début de *tweet* sert à interpeller ou répondre à un utilisateur alors que l'utilisation de @media en fin de *tweet* sert à accroître sa visibilité. Les relations n-aire sont aussi nécessaires pour décrire l'usage combiné des différents opérateurs, par exemple l'utilisation conjointe de RT et @ dans le même *tweet* ou pour détailler les objets contenus dans un *tweet*.

Dans la section 2, nous montrons que la modélisation classique des réseaux complexes sous forme de graphe simple ou de multi-graphe est insuffisante. Puis, en section 3, nous proposons et formalisons une modélisation sous la forme d'un réseau multiplexe hétérogène. Dans la section 4, nous détaillons l'algorithme PageRank puis une de ses extensions aux réseaux multiplexes. Dans la section 5, nous appliquons cet algorithme sur un jeu de données issu du projet pluridisciplinaire TEE'2014 – étude de la communication politique sur *Twitter* durant les élections européennes de 2014 – afin de déterminer l'influence des candidats, puis nous étudions les paramètres qui modifient le comportement de l'algorithme.

2. Modélisation des données des réseaux sociaux sous forme de graphe

Afin de représenter des réseaux complexes, des modélisations sous forme de graphe ont été proposées. Dans la section suivante, nous détaillons ces différentes modélisations, en prenant comme exemple *Twitter*.

Pour représenter les réseaux complexes tels que *Twitter* tout en ayant la possibilité de visualiser leurs différentes entités, plusieurs modèles de graphe ont été proposés allant du graphe simple à des formes plus évoluées.

2.1. Multi-graphe

Dans (Basaille *et al*., 2016), nous avons proposé une modélisation sous forme d'un **multi-graphe orienté avec labels** implémentée dans Neo4j. La figure 1 présente cette implémentation où les entités (*tweets*, utilisateurs, hashtags, etc.) sont les nœuds et les relations sont décrites par les liens. Les liens représentés par un pointillé sont déduits, les liens représentés par un trait plein sont des liens directs.

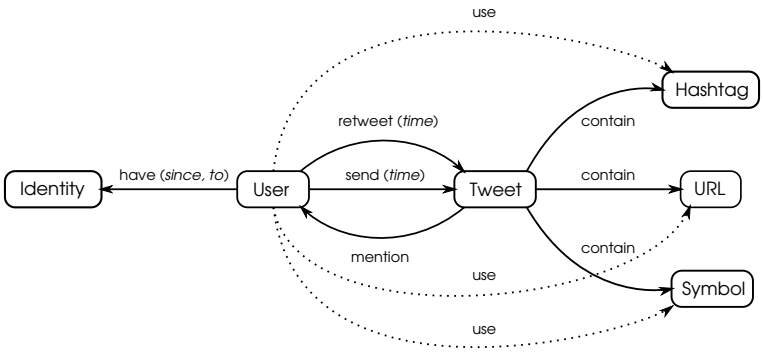


Figure 1. Représentation multi-graphe orienté avec labels de Twitter

La modélisation traditionnelle des réseaux complexes sous forme de graphe simple ou multi-graphe n'est pas suffisante. Par exemple, la relation *Mention* peut s'appliquer entre un *tweet* et un utilisateur mais aussi entre deux utilisateurs lorsque l'on agrège le nombre de mentions d'un utilisateur fait par l'utilisateur source ($Mention_U$) ou la relation $Mention_{ctx}$ qui agrège les relations *Écrire* et *Mention* (voir le tableau 1). Les relations n-aire sont aussi nécessaires pour décrire l'usage des différents opérateurs, par exemple l'utilisation conjointe de RT et @ dans le même *tweet* ou pour détailler les objets contenus dans un *tweet*, par exemple le *tweet* t_{1234} peut contenir une URL et deux hashtags. Un nouveau cadre plus général est alors fourni par les hypergraphes.

2.2. Hypergraphe

Un hypergraphe H est un couple (V, E) où V est un ensemble de nœuds et E est un ensemble d'hyperliens. Chaque hyperlien $e \in E$ peut contenir arbitrairement plusieurs nœuds, E est donc défini comme un sous-ensemble de 2^V. Cette modélisation permet de combiner des informations sur les utilisateurs et le contenu.

Twitter peut être modélisé comme un hypergraphe orienté dont V est l'ensemble des nœuds qui représentent les utilisateurs V_U, les *tweets* V_T et les objets V_O tels que les hashtags, les URLs et les symboles (emoji) [1] et un ensemble de relations R entre ces nœuds. Ces deux ensembles sont définis comme suit : $V = \{V_U \cup V_T \cup V_O\}$ et

1. Les intersections des ensembles pris deux à deux sont vides, c'est-à-dire $V_U \cap V_T = \varnothing$ et $V_U \cap V_O = \varnothing$ et $V_T \cap V_O = \varnothing$.

$R = \{R_i, i = 1, \ldots, m\}$ avec $R_i : V^p \rightarrow \mathbb{N}$. Le tableau 1 présente quelques relations possibles dans *Twitter*.

Tableau 1. Exemple de relations possibles dans Twitter

Relation	**Signature**
Écrire	*utilisateur* $\times$ *tweet* $\rightarrow \mathbb{B}$
Suivre	*utilisateur* $\times$ *utilisateur* $\rightarrow \mathbb{B}$
Retweet$_T$	*tweet* $\times$ *tweet* $\rightarrow \mathbb{B}$
Retweet$_U$	*utilisateur* $\times$ *utilisateur* $\rightarrow \mathbb{N}$
Retweet$_{ctx}$	*utilisateur* $\times$ *tweet* $\times$ *utilisateur* $\times$ *tweet* $\rightarrow \mathbb{B}$
Mention	*tweet* $\times$ *utilisateur* $\rightarrow \mathbb{B}$
Mention$_U$	*utilisateur* $\times$ *utilisateur* $\rightarrow \mathbb{N}$
Mention$_{ctx}$	*utilisateur* $\times$ *tweet* $\times$ (*utilisateur*)$^n \rightarrow \mathbb{B}$
Réponse	*tweet* $\times$ *tweet* $\rightarrow \mathbb{B}$
Réponse$_U$	*utilisateur* $\times$ *utilisateur* $\rightarrow \mathbb{N}$
Réponse$_{ctx}$	*utilisateur* $\times$ *tweet* $\times$ *utilisateur* $\times$ *tweet* $\rightarrow \mathbb{B}$
Contenir	*tweet* $\times$ *objet* $\rightarrow \mathbb{B}$
Contenir$_U$	*utilisateur* $\times$ *objet* $\rightarrow \mathbb{N}$
Contenir$_{ctx}$	*utilisateur* $\times$ *tweet* $\times$ (*objet*)$^n \rightarrow \mathbb{B}$

Prenons l'exemple d'un réseau *Twitter* comprenant les nœuds et relations décrites ci-dessous : u_1 à u_4 sont les utilisateurs, t_1à t_5 sont les tweets et h_1, h_2 représentent deux hashtags qui sont des objets particuliers.

R$_1$: *Suivre*: $(u_1, u_2) \rightarrow 1$ **R$_2$:** *Retweet*$_{ctx}$: $(u_1, t_1, u_2, t_2) \rightarrow 1$
R$_3$: *Mention*$_{ctx}$: $(u_2, t_2, u_3) \rightarrow 1$ **R$_4$:** *Mention*$_{ctx}$: $(u_2, t_5, u_3) \rightarrow 1$
R$_5$: *Mention*$_U$: $(u_2, u_3) \rightarrow 2$ **R$_6$:** *Réponse*$_{ctx}$: $(u_3, t_3, u_4, t_4) \rightarrow 1$
R$_7$: *Contenir*$_{ctx}$: $(u_4, t_4, h_1, h_2) \rightarrow 1$

Ce réseau est modélisé par l'hypergraphe donné en figure 2 avec :
$V = \{u_1, u_2, u_3, u_4, t_1, t_2, t_3, t_4, t_5, h_1, h_2\}$, $R = \{R_1, R_2, R_3, R_4, R_5, R_6, R_7\}$

Cette modélisation ne permet pas de représenter l'orientation des hyperliens mais l'hypergraphe peut être représenté par sa matrice d'incidence A_{ir} à coefficients entiers appartenant à l'ensemble $\{0, +1, -1\}$ telle que chaque colonne correspond à un hyperlien r, et chaque ligne à un nœud i de l'hypergraphe H. Si un hyperlien r arrive au nœud i, alors $A_{ir} = 1$, si un hyperlien r sort du nœud i, alors $A_{ir} = -1$, $A_{ir} = 0$ sinon (voir la figure 3).

Bien que les hypergraphes offrent un moyen de modéliser *Twitter* selon plusieurs points de vue sur une même relation, cette modélisation reste limitée puisque les hyperliens n'ont pas de sémantique explicite et que certaines informations sont perdues. Par exemple, les mentions peuvent être utilisées plusieurs fois dans un même *tweet*,

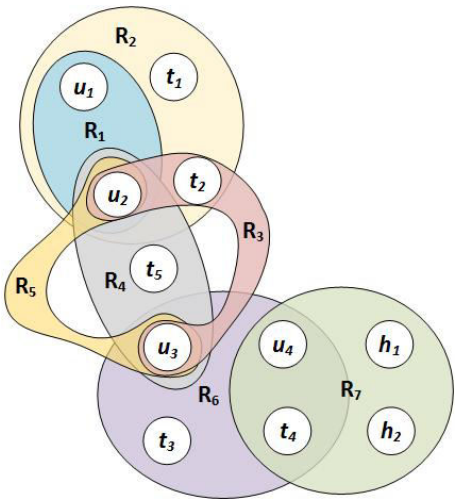


Figure 2. Exemple d'une modélisation de Twitter sous la forme d'un hypergraphe

ainsi l'arité [2] n'est pas fixée. Nous pouvons les modéliser par un ensemble de relations *Mention* ou $Mention_{ctx}$ mais l'ordre dans lequel les utilisateurs sont mentionnés est perdu. De plus, dans des réseaux complexes tels que *Twitter*, nous pouvons atteindre des milliers d'interactions, avec une telle modélisation, nous serons confrontés à plusieurs types d'hyperliens, ce qui peut compliquer la lisibilité des interactions. Par ailleurs, comparé au graphe simple, peu d'algorithmes ont été développés pour les hypergraphes même si dans (Estrada, Rodriguez-Velazquez, 2005) des mesures de centralité et le coefficient de clustering ont été étendus aux hypergraphes.

3. Modélisation des réseaux sociaux sous forme de réseaux multi-couches

Récemment, plusieurs travaux de recherche ont proposé d'améliorer et de généraliser les approches existantes pour prendre en compte des réseaux incluant plusieurs sous-systèmes et différents niveaux de connectivité entre les systèmes. Bien que la terminologie varie largement, en fonction de chaque cas spécifique, de tels réseaux peuvent généralement être appelés réseaux multi-couches.

3.1. Typologie des réseaux multi-couches

Les notations utilisées pour des graphes simples sont étendues pour permettre de représenter les structures qui ont des couches en plus des nœuds et des liens. Dans

2. L'arité d'une fonction est le nombre d'arguments. Par exemple, dans un *tweet* on peut utiliser n mentions n n'étant pas connu.

$$
\begin{array}{c} \\ u_1 \\ u_2 \\ u_3 \\ u_4 \\ t_1 \\ t_2 \\ t_3 \\ t_4 \\ t_5 \\ h_1 \\ h_2 \end{array}
\begin{array}{c}
\begin{array}{ccccccc} R_1 & R_2 & R_3 & R_4 & R_5 & R_6 & R_7 \end{array} \\
\left(\begin{array}{ccccccc}
-1 & -1 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & -1 & -1 & -1 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & -1 \\
0 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & -1 \\
0 & 0 & 0 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1
\end{array} \right)
\end{array}
$$

Figure 3. Matrice d'incidence de l'exemple donné en figure 2

(Bianconi, 2018), G. Bianconi propose une modélisation en considérant un réseau multi-couches comme un triplet $\mathcal{M} = (Y, \overrightarrow{G}, \mathcal{G})$ où

– Y est l'ensemble des couches, $Y = \{\alpha, \alpha \in \{1, 2, \ldots, M\}\}$ et M indique le nombre total de couches ($M = |Y|$) ;

– $\overrightarrow{G}$ une liste ordonnée de réseaux (graphes) caractérisant les interactions à l'intérieur de chaque couche $\alpha = 1, 2, \ldots, M$ soit $\overrightarrow{G} = (G_1, G_2, \ldots, G_\alpha, \ldots, G_M)$ pour lesquels $G_\alpha = (V_\alpha, E_\alpha)$ et $N_\alpha = |V_\alpha|$;

– $\mathcal{G}$ est une liste ordonnée de taille au plus $M \times M$ de graphes bi-parti caractérisant les interactions entre des paires de nœuds issues de couches différentes. Un élément $\mathcal{G}_{\alpha,\beta}$ de cette liste est défini par $\mathcal{G}_{\alpha,\beta} = (V_\alpha, V_\beta, E_{\alpha,\beta})$ pour $\alpha < \beta$.

Un nœud peut exister dans plusieurs couches et il apparaît dans au moins une couche. Ainsi, le n-uplet nœud-couches $(i, \alpha_1, \cdots, \alpha_M)$ indique si nœud i existe dans une couche $\alpha_j, j \in \{1, \ldots, M\}$.

Il existe plusieurs catégories de réseaux multi-couches, comme les réseaux multi-relationnels, hétérogènes, temporels, multiplexes. Les termes les désignant ont évolué au fur et à mesure de la publication des travaux de recherche. On peut remarquer que l'article de Kivelä (Kivelä *et al.*, 2014) permet une unification de la terminologie. Dans la suite de cette section, nous nous intéressons aux types réseaux multi-couches permettant de modéliser le réseau *Twitter*.

Dans un réseau multi-relationnel, les liens sont étiquetés avec un certain type, chaque type représente une relation spécifique entre les nœuds dans le réseau et se traduit en une couche. L'ensemble des liens E est divisé en classes disjointes : $E = \bigcup_{r \in R} E_r$, où R est l'ensemble de relations possibles. En conséquence, les couches sont implicites et dans chaque couche l'ensemble des nœuds est homogène, les liens également. Pour les réseaux sociaux tels que *Twitter* et afin de modéliser l'hétérogénéité des relations, (Dai *et al.*, 2012 ; Kivelä *et al.*, 2014) proposent une modélisation

sous la forme de réseau multi-relationnel. Cependant, cette modélisation n'est pas complètement satisfaisante car elle fixe une contrainte forte : l'ensemble des nœuds est homogène. Or, dans *Twitter*, les relations peuvent exister entre différents types de nœuds comme l'illustre le tableau 1.

Les réseaux hétérogènes ou réseaux multi-types possèdent des nœuds étiquetés avec un certain type qui peuvent être adjacents à des nœuds étiquetés avec le même type ou un type différent (Vazquez, 2006 ; Allard *et al*., 2009). *Twitter* peut être vu comme un réseau hétérogène puisque les nœuds peuvent avoir plusieurs types comme utilisateurs, *tweets*, hashtags, etc. Mais cette modélisation reste insuffisante puisque l'hétérogénéité des liens n'est pas prise en compte.

Dans un réseau temporel, chaque couche correspond à un évènement, à un timestamp ou un intervalle de temps. Ainsi, les réseaux temporels permettent de représenter un ensemble d'évènements consécutifs comme une séquence ordonnée de réseaux. Dans le cas d'un ensemble d'évènements, les évènements peuvent être représentés comme des triplets $e = (i, j, t)$ où $i, j \in V$ sont des nœuds et $t \in T$ est le timestamp d'un évènement. *Twitter* peut être modélisé comme un réseau temporel puisque chaque *tweet* possède un timestamp.

Un réseau multiplexe est une spécialisation de réseau multi-couches qui impose au moins un nœud en commun entre les couches et qui permet que les nœuds n'existent pas dans toutes les couches (Kanawati, 2015) (voir la figure 4). (Bianconi, 2018) propose une modélisation issue d'une simplification de la structure des réseaux multi-couches généraux respectant les propriétés suivantes :

– les réseaux multiplexes sont des réseaux multi-couches pour lesquels il existe une correspondance un à un entre les nœuds des différentes couches (nommés nœuds réplicas). Les correspondances peuvent être omises si tous les nœuds sont présents dans toutes les couches ;

– si il existe des liens entre les différentes couches (*interlinks*), ils expriment uniquement des correspondances entre nœuds réplicas.

Les réseaux multiplexes sont utilisés pour modéliser les réseaux sociaux en représentant différents types d'interactions pour le même ensemble d'utilisateurs. Ils peuvent aussi représenter des interactions entre différents types de nœuds en respectant la contrainte de correspondance un à un uniquement. La représentation des réseaux multiplexes sans liens inter-couches est similaire à celle des réseaux temporels : $\mathcal{M} = (Y, \overrightarrow{G})$. D'un point de vue algébrique, un réseau multiplexe pondéré et orienté peut être représenté par une suite de matrices d'adjacences $(A^{[\alpha]})_{\alpha \in \{1,2,\ldots,M\}}$.

Dans la plupart des réseaux multiplexes, les nœuds des différentes couches sont en correspondance 1-1 pour indiquer qu'il s'agit de la même entité. Dans certains cas, il peut être utile de distinguer l'identité des nœuds dans les différentes couches. On a alors recours à des liens inter-couches qui peuvent porter un poids spécifiant, comme dans les réseaux de transports, le coût associé à un changement de couche. Pour cela, on adopte une notation complémentaire pour les N nœuds et les M couches. Chaque nœud i peut avoir M réplicas représentés par des couples (i, α) avec $i = 1, 2, \ldots, N$

et $\alpha = 1, 2, \ldots, M$, $V_\alpha = \{(i, \alpha), i \in \{1, 2, \ldots, N\}\}$. Un réseau multiplexe avec des liens inter-couches est alors modélisé par un triplet $\mathcal{M} = (Y, \overrightarrow{G}, \mathcal{G})$ avec $\overrightarrow{G} \neq \varnothing$ et $\mathcal{G} \neq \varnothing$. Chaque couche α est représentée par un graphe $G_\alpha = (V_\alpha, E_\alpha)$ qui lui-même peut être représenté par sa matrice d'adjacence, $A^{[\alpha]}$ de taille $N \times N$ incluant éventuellement des liens dirigés et/ou pondérés. Le réseau de couplage représentant les liens entre les couches $\mathcal{G}_{\alpha,\beta} = (V_\alpha, V_\beta, E_{\alpha,\beta})$ connecte les nœuds réplicas avec les liens définis par $E_{\alpha,\beta} = \{((i, \alpha), (i, \beta)), i \in \{1, 2, \ldots, N\}\}$ et représenté par une matrice d'adjacence $\mathscr{C}^{[\alpha,\beta]}$, nommée alors matrice de couplage de taille $N \times N$, où les éléments non diagonaux sont nuls. Afin d'exploiter un réseau multiplexe, celui-ci doit être dans un format que les algorithmes sont capables de traiter directement ou bien dans un format à partir duquel il est possible de dériver les structures de données nécessaires aux algorithmes. Dans le cadre général des réseaux multi-couches, la notion de matrice de supra-adjacence est souvent utilisée. Elle peut être obtenue à partir des différentes matrices d'adjacence intra-couche (composante $\overrightarrow{G}$) et des matrices de couplage inter-couches (composante $\mathcal{G}$). La matrice de supra-adjacence de taille $(N \times M) \times (N \times M)$ $\mathscr{S}$, est définie de la manière suivante :

$$\mathscr{S}_{(i,\alpha),(j,\beta)} = \begin{cases} A_{ij}^{[\alpha]} & \text{si } \alpha = \beta \\ \mathscr{C}_{ij}^{[\alpha,\beta]} & \text{si } \alpha \neq \beta \end{cases}$$

avec $A^{[\alpha]}$ la matrice d'adjacence de la couche α et $\mathscr{C}^{[\alpha,\beta]}$ la matrice de couplage entre les couches α et β (voir figure 5).

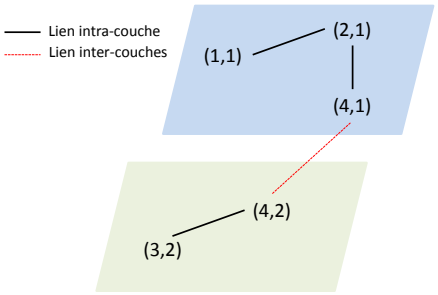


Figure 4. Exemple de réseau multiplexe ayant 4 nœuds et 2 couches

3.2. Modélisation de Twitter sous forme de réseau multiplexe hétérogène

Un modèle adapté aux données de *Twitter* est le modèle de réseau multiplexe puisque les différentes couches peuvent être exploitées pour représenter les différents types de relations. Or, ces relations peuvent exister entre différents types de nœuds (voir le tableau 1). Comme le modèle de réseaux multiplexes ne contraint pas les nœuds à avoir le même type, ni à tous exister dans toutes les couches, ce modèle est le plus pertinent. Dans le cadre de nos expérimentations, nous ne nous intéressons pas à l'aspect temporel, en revanche, nous souhaitons présenter les différentes entités

	$(1,1)$	$(2,1)$	$(3,1)$	$(4,1)$	$(1,2)$	$(2,2)$	$(3,2)$	$(4,2)$
$(1,1)$	0	1	0	0	0	0	0	0
$(2,1)$	1	0	0	1	0	0	0	0
$(3,1)$	0	0	0	0	0	0	0	0
$(4,1)$	0	1	0	0	0	0	0	1
$(1,2)$	0	0	0	0	0	0	0	0
$(2,2)$	0	0	0	0	0	0	0	0
$(3,2)$	0	0	0	0	0	0	0	1
$(4,2)$	0	0	0	1	0	0	1	0

Figure 5. Matrice de supra-adjacence de l'exemple donné en figure 4

du réseau telles que les *tweets* et les utilisateurs par différents types de nœuds, par conséquent, nous proposons de modéliser *Twitter* sous forme d'un réseau multiplexe hétérogène.

Nous spécialisons le modèle de réseau multiplexe général proposé par Bianconi (2018) et nous proposons un modèle pour *Twitter* défini par $\mathcal{M} = (Y, \overrightarrow{G}, \mathcal{G})$ où :

– Y est l'ensemble des couches correspondant aux différentes relations binaires du tableau 1 :

$$Y = \{\acute{E}crire, Suivre, Retweet_T, Retweet_U, Mention, Mention_U, \\ R\acute{e}ponse, R\acute{e}ponse_U, Contenir, Contenir_U\}$$

– $\overrightarrow{G}$ est la liste des graphes représentant les couches définies à partir de l'ensemble des nœuds $V = \{V_U \cup V_T \cup V_O\}$ pour lesquels V_U est l'ensemble des utilisateurs, V_T l'ensemble des tweets et V_O l'ensemble des objets c'est-à-dire un hashtag, une URL ou un symbole (emoji par exemple). Les graphes de $\overrightarrow{G}$ sont :

- $G_{\acute{E}crire} = (V_U \cup V_T, E_{\acute{E}crire})$ est un graphe orienté avec $E_{\acute{E}crire} \subseteq V_U \times V_T$;

- $G_{Suivre} = (V_U, E_{Suivre})$ est un graphe orienté avec $E_{Suivre} \subseteq V_U \times V_U$;

- $G_{Retweet_T} = (V_T, E_{Retweet_T})$ est un graphe orienté qui représente les retweets avec $E_{Retweet_T} \subseteq V_T \times V_T$;

- $G_{Retweet_U} = (V_U, E_{Retweet_U})$ est un graphe orienté pondéré avec $E_{Retweet_U} \subseteq V_U \times V_U$, dans ce graphe les liens représentent le nombre de retweets d'un utilisateur donné effectués par un autre ;

- $G_{Mention} = (V_U \cup V_T, E_{Mention})$ est un graphe orienté qui représente les mentions des utilisateurs dans les *tweets*, $E_{Mention} \subseteq V_U \times V_T$;

- $G_{Mention_U} = (V_U, E_{Mention_U})$ est un graphe orienté pondéré avec $E_{Mention_U} \subseteq V_U \times V_U$, dans ce graphe, les liens représentent le nombre de mentions d'un utilisateur dans les *tweets* émis par un autre utilisateur ;

- $G_{R\acute{e}ponse} = (V_T, E_{R\acute{e}ponse})$ est un graphe orienté qui représente les *tweets* qui sont des réponses à d'autres *tweets*, $E_{R\acute{e}ponse} \subseteq V_T \times V_T$;

- $G_{Réponse_U} = (V_U, E_{Réponse_U})$ est un graphe orienté pondéré avec $E_{Réponse_U} \subseteq V_U \times V_U$, dans ce graphe, les liens représentent le nombre de réponses d'un utilisateur à un autre, tous *tweets* confondus ;

- $G_{Contenir} = (V_T \cup V_O, E_{Contenir})$ est un graphe orienté qui représente les objets contenus (hashtag, emoji, URL, images) dans les *tweets*, $E_{Contenir} \subseteq V_T \times V_O$;

- $G_{Contenir_U} = (V_U \cup V_O, E_{Contenir_U})$ est un graphe orienté pondéré avec $E_{Contenir_U} \subseteq V_U \times V_O$, dans ce graphe, les liens représentent le nombre de fois où un utilisateur a utilisé un objet, tous *tweets* confondus.

– $\mathcal{G}$ représente les liens inter-couches modélisés par les trois types de graphes suivants :

- correspondances d'identités entre utilisateurs : $\{\mathcal{G}_{\alpha,\beta}\}$ avec α et $\beta \in \{Écrire, Suivre, Retweet_U, Mention, Mention_U, Réponse_U, Contenir_U\}$ sont vingt-et-un graphes non orientés ($(7 \times 6)/2$) qui expriment les correspondances 1-1 entre utilisateurs entre des couches dont un des types de nœud inclut V_U, $E_{\alpha,\beta} \subseteq V_U \times V_U$ avec comme contrainte $|E_{\alpha,\beta}| \leqslant |V_U|$;

- correspondances d'identités entre tweets : $\{\mathcal{G}_{\alpha,\beta}\}$ avec α et $\beta \in \{Écrire, Retweet_T, Mention, Réponse, Contenir\}$ sont dix graphes non orientés qui traduisent les correspondances 1-1 entre *tweets* dans les couches qui incluent des nœuds de type V_T, $E_{\alpha,\beta} \subseteq V_T \times V_T$ avec comme contrainte $|E_{\alpha,\beta}| \leqslant |V_T|$;

- correspondances d'identités entre objets : $\{\mathcal{G}_{\alpha,\beta}\}$ avec α et $\beta \in \{Contenir, Contenir_U\}$ est un graphe non orienté qui traduit les correspondances 1-1 entre objets dans les deux couches qui incluent des nœuds de type V_O, $E_{\alpha,\beta} \subseteq V_O \times V_O$ avec comme contrainte $|E_{\alpha,\beta}| \leqslant |V_O|$.

4. Influence et algorithme PageRank étendu au réseau multiplexe

Dans (Leavitt *et al*., 2009), l'influence dans le réseau social *Twitter* est définie comme la capacité d'un utilisateur à provoquer une action chez un autre utilisateur. Le terme « action » désigne les différentes relations possibles entre les nœuds (utilisateurs). Par conséquent, la mesure de l'influence dans *Twitter* est un problème complexe puisque *Twitter* offre plusieurs types de relations (*retweet*, *réponse*, *mention*, *suivre*), qui peuvent être combinés pour former différentes interactions.

Pour estimer l'influence d'un nœud, de nombreuses approches ont été proposées (Riquelme, González-Cantergiani, 2016 ; Al-Garadi *et al*., 2018) : certaines basées sur un résumé statistique de *Twitter*, d'autres sur la topologie du réseau avec différentes mesures de centralités ou encore en classant les nœuds avec des algorithmes basés sur le PageRank et le HITS, ou encore sur la fusion d'information afin de tenir compte de la diversité des liens. Certaines mesures de centralité et l'algorithme de PageRank ont été transposés pour le modèle des réseaux multiplexes. Solé-Ribalta *et al*. (2014) proposent une mesure de centralité d'intermédiarité qui prend en compte la structure inhérente des réseaux multiplexes et proposent un algorithme pour la calculer de manière efficace. Spatocco *et al*. (2018) ont présenté une méthodologie géné-

rale de calcul de mesures de centralité de manière itérative, basée sur le théorème de Perron-Frobenius [3], permettant les classements sur une série de matrices représentant un réseau multiplexe. Dans la suite, nous présentons le PageRank et un algorithme de PageRank étendu au réseau multiplexe nommé Multiplex PageRank.

4.1. PageRank

L'idée principale du PageRank est que « Les pages les plus importantes (des sites Web) sont susceptibles de recevoir plus de liens à partir d'autres pages » (Page *et al*., 1999). PageRank suppose que l'importance d'une page Web est déterminée par la quantité et la qualité des pages qui s'y rattachent. Initialement, chaque nœud (c'est-à-dire, page) reçoit une valeur PR. Ensuite, chaque nœud distribue uniformément sa valeur PR à ses voisins à travers les liens sortants. Mathématiquement, la valeur PR du nœud i à l'étape t est :

$$PR_i(t) = \sum_{j=1}^{n} a_{ji} \frac{PR_j(t-1)}{k_j^{out}}$$

où n est le nombre total de nœuds dans le réseau, A est la matrice d'adjacence et k_j^{out} est le degré de centralité sortant du nœud j. L'itération ci-dessus s'arrêtera si les valeurs PR de tous les nœuds atteignent un état stationnaire. Un inconvénient majeur du processus de marche aléatoire du PR ci-dessus est que la valeur PR d'un nœud pendulaire (c'est-à-dire, un nœud avec zéro pour out-degree) ne peut pas être redistribuée, ainsi la formule ne peut pas garantir la convergence. Pour résoudre ce problème, un facteur de téléportation a été introduit en supposant que l'utilisateur navigue entre les pages Web en suivant les liens avec la probabilité s, et quitte la page actuelle pour une page aléatoire avec une probabilité de $1 - s$, $s \in [0, 1]$ est généralement fixé empiriquement autour de $0,85$. En conséquence, la formule du PR est modifiée comme suit :

$$PR_i(t) = s \sum_{j=1}^{n} a_{ji} \frac{PR_j(t-1)}{k_j^{out}} + (1-s)\frac{1}{n}$$

4.2. Multiplex PageRank

Des auteurs ont proposé une modification de l'algorithme PageRank pour l'adapter aux réseaux multiplexes. Halu *et al*. (2013) décrivent le pageRank multiplexe sur un réseau multiplexe ayant deux couches, Iacovacci et Bianconi (2016) introduisent en plus une fonction de similarité entre couches. Nous reprenons cet algorithme, et nous l'adaptons à notre formalisation du réseau multiplexe décrivant *Twitter*. Le principe

3. http://www.bibmath.net/dico/index.php?action=affiche&quoi=./p/perron-frobenius.html

de base consiste à choisir une couche principale α avec sa matrice d'adjacence $A^{[\alpha]}$ et à calculer le PageRank $x_i^{[\alpha]}$, $i = 1, \ldots, N_\alpha$ pour chaque nœud présent dans cette couche par la formule habituelle soit :

$$x_i^{[\alpha]} = d_\alpha \sum_{j=1}^{N_\alpha} A_{ji}^{[\alpha]} \frac{x_j^{[\alpha]}}{g_j} + (1 - d_\alpha)\frac{1}{N_\alpha}$$

où d_α est le facteur de téléportation de la couche α, $g_j = max(1, \sum_{r=1}^{N_\alpha} A_{jr})$.

L'ensemble des valeurs constitué par chaque $x_i^{[\alpha]}$ est représenté par le vecteur $X^{[\alpha]}$ pour la couche α. Ensuite, l'algorithme PageRank multiplexe mesure une forme de centralité pour tous les nœuds (i, β) présents dans une autre couche β en utilisant la matrice d'adjacence $A^{[\beta]}$ et prenant en compte les mesures effectuées sur la couche α. Les mesures sur la couche β sont influencées par le voisinage du nœud (i, β) et par les nœuds réplicas dans la couche α. On obtient ainsi une formule générale du PageRank :

$$x_i^{[\beta]} = d_\beta \sum_{j=1}^{N_\beta} x_i^{[\alpha]} A_{ji}^{[\beta]} \frac{x_j^{[\beta]}}{G_j} + (1 - d_\beta)\frac{1}{N_\beta}\frac{x_i^{[\alpha]}}{\langle X^{[\alpha]} \rangle} \tag{1}$$

où $G_j = \sum_{r=1}^{N_\beta} A_{jr}^{[\beta]} x_r^{[\alpha]} + \delta(0, \sum_{r=1}^{N_\beta} A_{jr}^{[\beta]} x_r^{[\alpha]})$ et $\delta(a, b)$ est la fonction Kronecker, c'est une fonction à deux variables qui est égale à 1 si celles-ci sont égales, et 0 sinon.

Le nœud (i, β) est directement influencé par son réplica (i, α), cette influence est prise en compte en biaisant le premier terme de l'équation du PageRank au niveau du nœud lui-même et de ses voisins. La contribution de chaque voisin (j, β) de (i, β) est atténuée en divisant la centralité de (j, β) par la somme des centralités que les voisins de (j, β) ont dans la couche α.

Le second terme de l'équation (1) traduit la contribution de la centralité du nœud (i, α) au nœud (i, β). En effet, même si un nœud de β n'a pas la capacité à attirer des voisins importants, il peut quand même bénéficier de sa centralité acquise dans la couche α. Une autre manière de formuler cette propriété est : l'importance d'un nœud dans une couche est affectée de manière bénéfique par l'importance qu'il possède dans une autre couche indépendamment de sa capacité à attirer d'autres nœuds. Pour ce faire on ajoute un facteur multiplicateur sur la partie de la formule liée au saut aléatoire qui est la valeur du PageRank de (i, α) dans la couche α divisée par la moyenne des valeurs des PageRank des nœuds de la couche α soit $\frac{x_i^{[\alpha]}}{\langle X^{[\alpha]} \rangle}$.

Ainsi, les effets de l'interaction entre les différentes couches du réseau sur la centralité de type PageRank sont directement pris en compte dans la marche aléatoire. En particulier, en fonction de l'intensité de l'interaction entre les couches, Iacovacci

et Bianconi (2016) définissent différentes versions du PageRank multiplexe : additif, multiplicatif et combiné.

$$x_i^{[\beta]} = d_\beta \sum_j^{N_\beta} (x_i^{[\alpha]})^b A_{ji}^{[\beta]} \frac{x_j^{[\beta]}}{G_j} + (1 - d_\beta) \frac{1}{N_\beta} \left(\frac{x_i^{[\alpha]}}{\langle X^{[\alpha]} \rangle} \right)^a \quad (2)$$

où $G_j = \sum_{r=1}^{N_\beta} A_{jr}^{[\beta]} \left(x_r^{[\alpha]} \right)^b + \delta(0, \sum_{r=1}^{N_\beta} A_{jr}^{[\beta]} \left(x_r^{[\alpha]} \right)^b)$

Les valeurs de a et b permettent d'ajuster les deux parties de la formule. On peut identifier quatre cas limites dont trois correspondent au PageRank multiplexe (Halu *et al*., 2013) puisque la cas $(a = 0, b = 0)$ correspond à un PageRank simple.

– PageRank multiplexe additif $(a = 1, b = 0)$, dans ce cas, $G_j = max(1, \sum_{r=1}^{N_\beta} A_{jr}^{[\beta]})$. Le premier terme pondère la centralité des nœuds de la couche β en fonction de la centralité de leurs voisins et le deuxième terme permet à chaque nœud de la couche β de tirer un avantage supplémentaire du fait qu'il est central dans la couche α, quelle que soit la pertinence des nœuds qui le désignent dans la couche β.

– PageRank multiplexe multiplicatif $(a = 0, b = 1)$, l'effet de la couche α se produit sur le voisinage des nœuds de la couche β. Ici, tous les avantages d'être central dans la couche α dépendent des connexions que le nœud reçoit des nœuds centraux de la couche β.

– PageRank multiplexe combiné $(a = 1, b = 1)$ ajoute les deux effets précédents.

5. Application à l'étude de l'influence sur Twitter

Bianconi fournit une fonction Matlab pour calculer le PageRank multiplexe [4]. Nous avons utilisé cette fonction avec les données du projet TEE'2014 dont l'objectif global est d'observer et d'analyser la communication des politiques sur *Twitter* durant les élections européennes de mai 2014. Nous évaluons l'influence des candidats *via* leurs scores PageRank dans un réseau multiplexe où chacune des trois relations *Retweet*, *Mention* et *Réponse* représente une couche. Afin de construire les couches, nous considérons les candidats et les utilisateurs ayant interagi avec eux selon ces trois relations. Les couches sont représentées par leur matrice d'adjacence, le facteur de téléportation utilisé a été fixé à $0,85$. Le tableau 2 résume les données utilisées.

Dans le cadre du PageRank multiplexe, la couche *Retweet* est choisie pour être la couche principale puisqu'elle est plus significative en terme d'influence comme l'a indiqué Cha *et al*. (2010). C'est à partir de cette couche que les autres calculs sont établis dans les trois cas : PageRank multiplexe multiplicatif, additif et combiné. La figure 6

4. https://github.com/ginestrab/Multiplex-PageRank

Tableau 2. Paramètres des données du corpus TEE'2014

Nombre de comptes	73 264	**Nombre de candidats**	616
Nombre de relations	614 191	**Nombre de *retweets***	145 633
Nombre de *mentions*	434 207	**Nombre de *réponses***	34 351

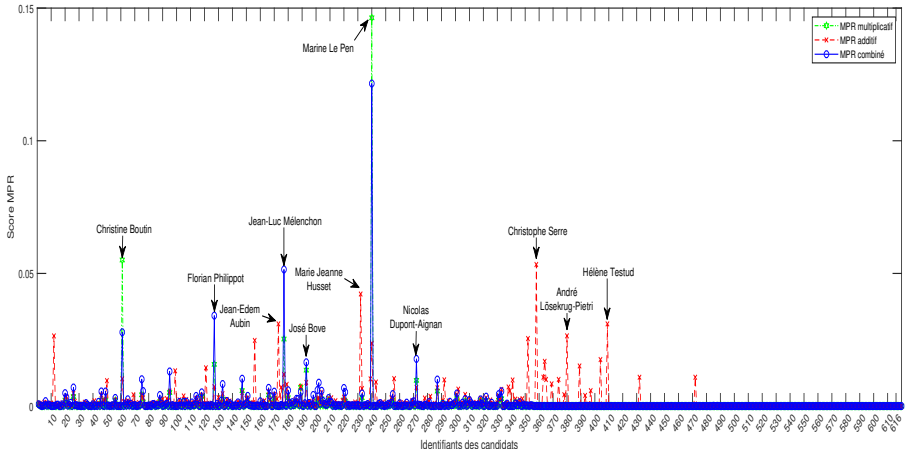


Figure 6. PageRank multiplexe multiplicatif, additif et combiné des candidats français

synthétise les scores PageRank pour les trois versions du PageRank multiplexe. À des fins de comparaison, nous avons aussi calculé le PageRank simple selon chacune des relations choisies (voir la figure 7). Nous observons que sur les dix premiers candidats six candidats sont en communs.

Les résultats obtenus ont été appréciés par nos collègues en Sciences Humaines et Sociales (SHS) participant au projet. Les PageRank multiplexe multiplicatif et combiné sont capables de distinguer correctement les personnes qui ont effectivement une influence importante dans le réseau *Twitter* et dans la vie réelle. Seuls les résultats du PageRank multiplexe additif sont très différents des autres résultats, faisant apparaître des "petits" candidats dont le cercle d'influence est plus restreint par rapport aux autres utilisateurs trouvés par les autres variations du PageRank multiplexe. Ces résultats mettant en avant de tels candidats sont certainement les plus intéressants pour nos collègues de SHS car ils présentent des singularités. Notre rôle d'informaticien n'est pas d'analyser les résulats mais de garantir leur qualité (nettoyage des données, utilisation correcte des algorithmes, etc.).

Cependant, bien que les PageRanks multiplexes permettent d'obtenir un bon classement d'utilisateurs, le score obtenu pour chaque utilisateur pris individuellement n'indique pas son influence et il est inexploitable si on ne le compare pas avec le score des autres utilisateurs.

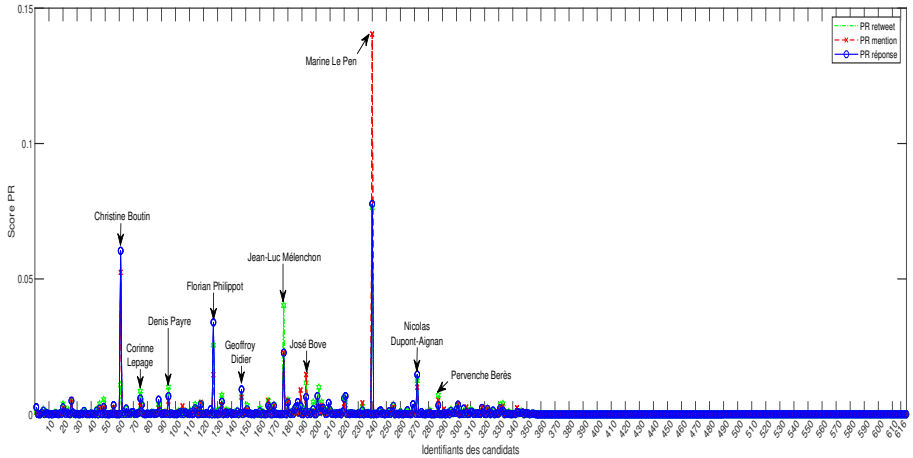


Figure 7. PageRank des candidats français selon les relations Retweet, Mention et Réponse

6. Conclusion

Dans cet article, nous avons proposé une nouvelle modélisation de *Twitter* sous forme d'un réseau multiplexe hétérogène. Nous avons formalisé et spécialisé ce modèle pour spécifier les relations engendrées par les interactions entre les utilisateurs de *Twitter* et issues de l'utilisation des différents opérateurs. Cette modélisation permet de visualiser, à travers les couches du réseau, les différentes relations présentes dans le réseau *Twitter*, les nœuds hétérogènes permettent à leur tour de présenter les différents types de nœuds tels que les utilisateurs et les *tweets*. Cette diversité de liens et de nœuds est primordiale afin d'étudier l'influence des utilisateurs de *Twitter* car les relations ne représentent pas la même importance dans l'estimation de l'influence. Ainsi, il est important de combiner les informations des différentes relations. Enfin, nous avons exploité ce réseau en utilisant une extension du PageRank pour les réseaux multiplexes. Nous avons appliqué cette extension aux données du projet TEE'2014 et réalisé différentes expériences en faisant varier les paramètres qui modifient le comportement de l'algorithme.

Bibliographie

Al-Garadi M. A., Varathan K. D., Ravana S. D., Ahmed E., Mujtaba G., Khan M. U. S. *et al.* (2018, janvier). Analysis of online social network connections for identification of influential users: Survey and open research issues. *ACM Comput. Surv.*, vol. 51, n° 1, p. 16:1–16:37. Consulté sur http://doi.acm.org/10.1145/3155897

Allard A., Noël P.-A., Dubé L. J., Pourbohloul B. (2009). Heterogeneous bond percolation on multitype networks with an application to epidemic dynamics. *Physical Review E*, vol. 79, n° 3, p. 036113.

Basaille I., Kirgizov S., Leclercq E., Savonnet M., Cullot N. (2016). Towards a twitter observatory: A multi-paradigm framework for collecting, storing and analysing tweets. In *Tenth IEEE international conference on research challenges in information science, (RCIS)*, p. 1–10.

Bianconi G. (2018). *Multilayer networks: Structure and function*. Oxford University Press.

Cha M., Haddadi H., Benevenuto F., Gummadi P. K. (2010). Measuring user influence in twitter: The million follower fallacy. *ICWSM*, vol. 10, n° 30, p. 10-17.

Dai B. T., Chua F. C. T., Lim E.-P. (2012). Structural Analysis in Multi-Relational Social Networks. In *Proceedings of the siam international conference on data mining*, p. 451-462.

Estrada E., Rodriguez-Velazquez J. A. (2005, 06). Complex Networks as Hypergraphs.

Halu A., Mondragón R. J., Panzarasa P., Bianconi G. (2013). Multiplex pagerank. *PloS one*, vol. 8, n° 10, p. e78293.

Iacovacci J., Bianconi G. (2016). Extracting information from multiplex networks. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 26, n° 6, p. 065306.

Kanawati R. (2015). Multiplex Network mining: a brief survey. *IEEE Intelligent Informatics Bulletin*, vol. 16, n° 1, p. 24–27.

Kivelä M., Arenas A., Barthelemy M., Gleeson J. P., Moreno Y., Porter M. A. (2014). Multilayer networks. *Journal of complex networks*, vol. 2, n° 3, p. 203–271.

Leavitt A., Burchard E., Fisher D., Gilbert S. (2009). The Influentials: New Approaches for Analyzing Influence on Twitter. *Webecology Project*.

Page L., Brin S., Motwani R., Winograd T. (1999). The PageRank citation ranking: Bringing order to the Web. In *Proceedings of the 7th international world wide web conference*, p. 161–172.

Riquelme F., González-Cantergiani P. (2016). Measuring user influence on twitter: A survey. *Information Processing & Management*, vol. 52, n° 5, p. 949–975.

Solé-Ribalta A., De Domenico M., Gómez S., Arenas A. (2014). Centrality Rankings in Multiplex Networks. In *Proceedings of the 2014 acm conference on web science*, p. 149–155. New York, NY, USA, ACM. Consulté sur http://doi.acm.org/10.1145/2615569.2615687

Spatocco C., D'Andrea A., Domeniconi C., Stilo G. (2018). A New Framework for Centrality Measures in Multiplex Networks. *CoRR*, vol. abs/1801.08026. Consulté sur http://arxiv.org/abs/1801.08026

Vazquez A. (2006). Spreading dynamics on heterogeneous populations: multitype network approach. *Physical Review E*, vol. 74, n° 6, p. 066114.

Génération automatique d'alternatives de structuration de données pour systèmes orientés document

Paola Gómez [1], **Rubby Casallas** [2], **Claudia Roncancio** [1]

1. *Univ. Grenoble Alpes, CNRS, Grenoble INP* [*], *LIG, 38000 Grenoble, France*
   {paola.gomez-barreto,claudia.roncancio}@univ-grenoble-alpes.fr
2. *TICSw, Universidad de los Andes, Bogotá - Colombia*
   rcasalla@uniandes.edu.co

RÉSUMÉ. *Les possibilités de structuration des données dans les systèmes orientés document sont très nombreuses même lorsque les données à traiter sont simples. Le choix de la structuration est néanmoins très important car il conditionne des aspects cruciaux de la base et des applications. Nos travaux visent à aider l'utilisateur à comprendre, évaluer et choisir, la structuration des données d'une base orientée document selon les besoins. Dans cet article, nous proposons de générer automatiquement des alternatives de structuration orientées document, à partir d'un schéma UML, afin de pouvoir les comparer. Nous adoptons une approche originale de génération inspirée des lignes de produits logiciels. Dans notre cas, le produit est une structure de données orientée document basée sur JSON. Nous utilisons des modèles de caractéristiques pour représenter les variantes et les points communs entre les produits tout en contrôlant l'explosion du nombre de combinaisons. Cet article présente l'approche, les algorithmes de génération et le prototype qui permet aux utilisateurs d'apprécier diverses alternatives de structuration pour une base de documents.*

ABSTRACT. *The possibilities of data structuring in document-oriented systems are numerous even for simple data. The choice of the structuring is nevertheless very important because of its impact on crucial aspects of the base and the applications. Our work aims to help users to understand, evaluate and choose the data structuring of a document-oriented database. In this paper, we propose to automatically generate document-oriented structuring alternatives from a UML model to compare them. We adopt an original approach of generation inspired by Software Product Lines. In our case, the product is a document-oriented structure based on JSON. We use feature models to represent variations and common points between products by controlling the explosion of combinations. This paper presents the approach, the generation algorithms and the prototype that allows users to appreciate data structuring alternatives.*

MOTS-CLÉS : *NoSQL, systèmes orientés document, variabilité, modèle des caractéristiques*

KEYWORDS: *NoSQL, document-oriented systems, variability, feature models*

*. Institute of Engineering Univ. Grenoble Alpes

1. Introduction

La flexibilité des modèles semi-structurés supportés par les systèmes NoSQL orientés documents ouvre la porte à de nombreuses possibilités de représentation des données. Ces possibilités peuvent être nombreuses et des alternatives de structuration potentiellement intéressantes peuvent être écartées car, par exemple, le développeur n'a pas les moyens d'analyser toutes les options, ou parce que, involontairement, un facteur clé n'est pas pris en compte. Ces alternatives sont difficiles à envisager, à comprendre et à gérer à cause du nombre et de l'absence de schéma de base de données dans des systèmes tels que MongoDB. Néanmoins le choix de la structuration de données a un impact important sur la base et les applications. Cela concerne, entre autres, les performances, la facilité de programmation et la facilité à comprendre, maintenir et faire évoluer le système (Gómez *et al*., 2016).

Nos propositions visent à aider le développeur à mener une phase de conception et d'analyse malgré les nombreuses alternatives possibles et l'utilisation d'un système sans schéma (*Schemaless*). Ainsi, nous explicitons le type des structures de données utilisées dans la base. Pour cela nous proposons le format AJSchéma, simple et lisible, qui sans jouer le rôle de schéma dans la base, permettra de mettre en évidence les types des données et ainsi de raisonner sur la structure.

Nous avons proposé l'approche SCORUS (Gomez, 2018) où le développeur modélise ses données à l'aide d'UML. Ce modèle est traité afin de générer un ensemble d'*alternatives de structuration orientées documents* sous forme d'AJSchémas. Ensuite, une phase d'évaluation, SCORUS évalue automatiquement des métriques pour chaque AJSchéma. Ces métriques constituent des indicateurs objectifs des caractéristiques des structures. Une phase d'analyse complète l'approche en s'appuyant sur les métriques et les préférences des utilisateurs tel que présenté dans (Gómez *et al*., 2018b ; 2018a). Le prototype ScorusTool implémente cette approche.

Cet article porte sur la phase de génération de variantes de structuration orientées document. Cette génération automatique permet à l'utilisateur d'apprécier facilement de multiples choix de structures dont l'analyse peut être poursuivie ou non. Pour ce générateur nous avons suivi une approche issue du domaine des lignes de produits logiciels. Il s'agit de l'approche par modèles de caractéristiques qui permet d'exprimer la variabilité entre diverses alternatives d'un produit (Kang *et al*., 2002). Ici, nous produisons diverses alternatives d'AJSchéma. Nous utilisons le modèle UML et les possibilités de structuration orientées documents afin d'identifier les alternatives possibles. Cela nécessite de formaliser les variations et points communs à travers d'un modèle de caractéristiques. La stratégie des modèles de caractéristiques permet de définir la variabilité des structures possibles d'une manière compacte et de contrôler l'explosion des possibilités qui peuvent survenir. Cet article présente l'approche et l'algorithme *AMISS* qui crée un modèle de caractéristiques contenant les variations des structurations. Pour chaque alternative de structuration fournie par ce modèle, nous dérivons les structures correspondantes dans divers formats dont les AJSchémas.

Dans la suite, la section 2 introduit un exemple illustrant la flexibilité et la variabilité des structures. La section 3 présente les principes des lignes de produits et des modèles de caractéristiques. La section 4 détaille les étapes de la ligne de produits basée

sur les modèles de caractéristiques qui permettent d'obtenir a partir d'un modèle UML des alternatives de structuration orientée document. Dans la section 5 nous introduisons l'algorithme *AMISS*. La section 6 introduit le prototype *ScorusTool*. Les travaux connexes sont décrits en section 7. Nos conclusions et perspectives de recherche sont présentées en section 8.

2. Flexibilité et évolution dans la modélisation semi-structurée

Nous sommes intéressés par la flexibilité d'une modélisation des données dans une base orientée document semi-structurée et ayant un grand nombre d'alternatives de structuration possibles. Dans cette section, nous illustrons ces aspects à travers un exemple, dans un contexte de marketing et en considérant un modèle conceptuel UML.

Dans les bases orientées document, les données sont gérées comme un ensemble de collections de documents. Un document est simplement un ensemble de paires `attribut:valeur`. Le type des valeurs peut être atomique ou complexe. Par complexe nous entendons soit un tableau de valeurs de tout type ou un document qui dans ce cas est dit *imbriqué*. Notons que la valeur d'un attribut peut être l'identifiant d'un document ou la valeur d'un attribut d'un document d'une autre collection. Cela permet de *référencer* un ou plusieurs documents. Ce système de types permet beaucoup de flexibilité dans la création des structures.

Nous considérons l'information des agences et des secteurs d'activités introduite par la Figure 1. La classe `Agency` représente une agence qui peut gérer plusieurs secteurs d'activités, représentés par la classe `Business` avec le rôle `bLines`. À son tour, un secteur d'activités est géré par une seule agence (rôle `ag` de l'association).

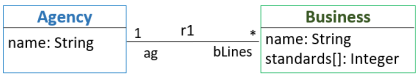


Figure 1. Modèle de classes pour des agences et des secteurs d'activités

Il y a plusieurs façons de modéliser ces données dans une base orientée documents. Chacune des classes peut inspirer la création d'une collection dont les documents contiennent l'ensemble des attributs de sa classe. Dans notre exemple, une collection `Agencies` (respectivement `Business`) correspondra à une collection d'agences dont les documents contiendront les attributs de la classe `Agency` (resp. `Business`).

Concernant l'association, elle peut se matérialiser soit par imbrication soit par référencement des informations de ses classes extrémités en considérant les deux sens de l'association. Cela introduit plusieurs façons de structurer les collections. Il s'agit alors d'avoir un ensemble d'options de manière à disposer de choix intéressants pour le concepteur tout en contrôlant le nombre de possibilités offertes. Nous définissons des **contraintes de structuration** qui portent sur la complétude et certaines formes de liens entre les données qui évitent l'isolement et une complexité inutile qui peut s'avérer coûteuse.

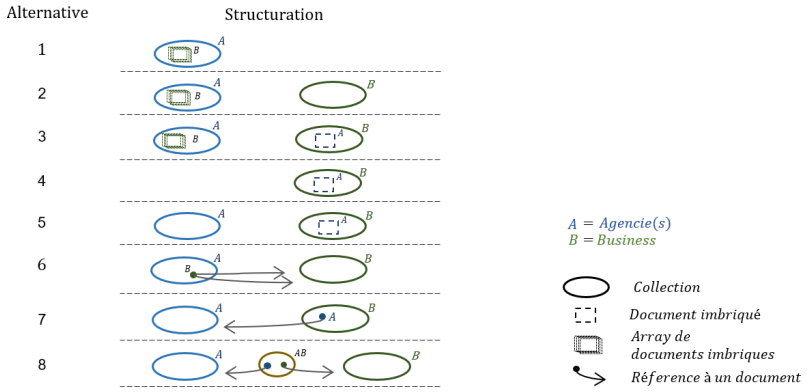


Figure 2. Alternatives de semi-structuration pour le modèle UML de la Figure 1

Existence : pour garantir l'existence de l'information, les alternatives de structuration devront inclure une collection si ses documents sont référencés.

Isolement : empêcher l'isolement potentiel

– d'une collection A qui n'imbrique pas l'information de sa collection partenaire B. A doit être référencée, ou imbriquée dans une autre collection.

– de deux collections. Considérer une *collection-lien* ne contenant que des références aux deux collections si aucune d'elles n'imbrique ni ne référence les informations de sa partenaire.

Références circulaires: pour limiter une forte complexité induite par des références circulaires, nous considérons les alternatives suivantes,

– empêcher les collections d'être référencées entre elles et

– si une collection A imbrique l'information d'une collection B alors, empêcher que cette information inclut une référence vers A.

Reprenons l'exemple de la Figure 1 au vu des contraintes de structuration énoncées. La représentation de l'association r_1 peut être faite selon les huit alternatives illustrées sur la Figure 2. Les documents d'une collection sont homogènes dans leur structure.

Les alternatives 6 et 7 respectent l'existence de la collection référencée. Les alternatives 2 et 5 empêchent l'isolement d'une collection en imbriquant et en dupliquant leur information. L'alternative 8 utilise une *collection-lien* pour les références. Les alternatives qui ne respectent pas les *contraintes de structuration* sont écartées, par exemple l'alternative avec les collections `Agencies` et `Businness` sans imbrication ni référencement entre elles.

Dans le cas où les besoins changent et de nouvelles classes sont ajoutées au modèle UML, la structure de stockage doit changer. De nouvelles alternatives de structuration sont possibles et cela peut impliquer la mise à jour de celles déjà existantes. Le nombre d'alternatives de structuration possibles peut être très grand et le choix difficile. Des contraintes sont nécessaires pour identifier des alternatives de structuration valides tout en limitant la combinatoire.

Notre contribution vise à assister les développeurs dans leurs choix de structuration de données. Pour cela nous proposons un générateur qui produit un ensemble d'alternatives de structuration potentiellement intéressantes. Nous avons analysé les différences et les points communs entre les structures possibles et les représentons à l'aide de modèles de caractéristiques. Cette stratégie permet d'exprimer les variantes entre des produits qui, dans notre cas, sont les alternatives de structuration. Dans la suite nous présentons les principes des lignes de produits logiciels qui sont adaptés pour notre générateur de structures.

3. Variabilité et modèles de caractéristiques

Dans les lignes de produits logiciel, il est nécessaire d'exprimer les différences entre les produits à construire. Il existe de nombreuses stratégies pour représenter cette variabilité. La plus utilisée est celle basée sur des modèles de caractéristiques (Kang *et al.*, 1990), dont l'objectif est de modéliser les combinaisons et les différences possibles entre les produits. On distingue les étapes de modélisation, configuration et dérivation, décrites ci-après.

3.1. Modélisation

La variabilité est représentée par un modèle de caractéristiques désigné par fm. Il est composé d'un *ensemble de caractéristiques* $\mathscr{F}$ et de contraintes qui forcent ou interdisent que deux ou plusieurs caractéristiques soient compatibles ou non.

DÉFINITION 1. — *Un Modèle de caractéristiques est défini comme un quadruplet*

$$fm = (\mathscr{F}, \mathscr{L}, rc, \mathscr{D})$$

où, $\mathscr{F}$ *est l'ensemble de caractéristiques,*
$\mathscr{L}$ *représente les liens qui relient les caractéristiques,*
rc *est la caractéristique racine,* $rc \in \mathscr{F}$*,*
$\mathscr{D}$ *est l'ensemble des contraintes entre les caractéristiques de la forme* $X \implies Y$

Les caractéristiques sont organisées dans une structure arborescente permettant de former des groupes de caractéristiques. La Figure 3 illustre les types de liens qui permettent de désigner les variations entre les caractéristiques :

– Une caractéristique *obligatoire* doit être choisie si le parent est choisi. Les caractéristiques obligatoires sont présentes dans toutes les configurations.
– Une caractéristique *optionnelle* peut être choisie si le parent est choisi. Si ni elle ni un de ses fils n'est choisi, la branche dont elle est racine est enlevée.
– Dans un *groupe OR*, une ou plusieurs caractéristiques du groupe doivent être choisies si le parent du groupe est choisi.
– Dans un *groupe XOR*, une unique caractéristique du groupe doit être choisie si le parent du groupe est choisi.

La Figure 4 développe un modèle de caractéristiques pour une *Liseuse*, nommé $fm_{liseuse}$. La caractéristique racine *Liseuse* a deux sous-caractéristiques obligatoires,

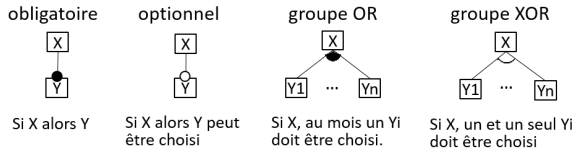


Figure 3. Liens entre les caractéristiques d'un modèle de caractéristiques

Écran et *Dispositif d'entrée*, et une optionnelle, *Son*. À son tour, la caractéristique *Écran*, implique le choix entre un écran *Tactile* ou un écran *Normal*. Par rapport au *Dispositif d'entrée*, il est possible de choisir soit le *Clavier*, soit le *Stylet* soit les deux. Le modèle $fm_{liseuse}$ considère deux contraintes, $\mathscr{D}(fm_{liseuse})$:

– forcer la sélection d'un clavier dans le cas où le son est choisi et,
– empêcher la sélection d'un dispositif d'entrée *Stylet* si un écran normal est choisi.

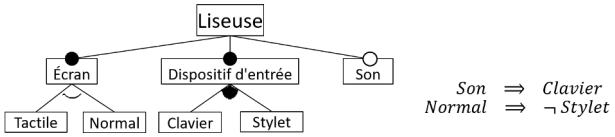


Figure 4. Modèle de caractéristiques $fm_{liseuse}$ pour une Liseuse

3.2. Processus de configuration

Une fois créé le modèle de caractéristiques fm, l'étape de configuration consiste à sélectionner des combinaisons valides entre ses caractéristiques, à savoir la sémantique du modèle notée $\|fm\|$. Une configuration valide est conforme aux contraintes et sera composée d'un sous-ensemble contenant les noms des caractéristiques choisies.

DÉFINITION 2. — *Configuration*
Une configuration C pour un modèle $fm = (\mathscr{F}, \mathscr{L}, rc, \mathscr{D})$ est définie comme un ensemble de caractéristiques sélectionnées parmi l'ensemble des caractéristiques du modèle $\mathscr{F}$. Une configuration est valide si et seulement si elle est conforme aux contraintes $\mathscr{D}$ définies dans le modèle.

$$C(fm) = \{f | f \in \mathscr{F}(fm)\}$$

Dans notre exemple, la sémantique $\|fm_{liseuse}\|$ serait composée de 128 ($2^{7_{caractéristiques}}$) configurations sans prendre en compte les contraintes. En les considérant, l'ensemble est réduit à 7 configurations valides, parmi lesquelles :

$$C_1(fm_{liseuse}) = \{\text{Liseuse}, \text{Ecran}, \text{Dispositif d'entrée}, \text{Tactile}, \text{Stylet}\}$$

$$C_2(fm_{liseuse}) = \{\text{Liseuse}, \text{Ecran}, \text{Dispositif d'entrée}, \text{Son}, \text{Tactile}, \text{Clavier}\}$$

3.3. Processus de dérivation

Une fois l'ensemble de configurations identifié, chacune est interprétée pour créer une version du produit correspondant aux caractéristiques choisies pour celle-ci.

La figure ci-contre illustre deux des 7 produits finaux du modèle de caractéristiques de la liseuse, noté $\mathscr{P}(fm_{liseuse})$. La dérivation de la configuration C_1 permet de créer une liseuse tactile avec un stylet alors que le produit correspondant à C_2 aura un clavier et une entrée sonore.

4. Génération d'alternatives de structuration orientées document

Nos travaux adoptent l'approche des lignes de produits pour contribuer à la qualité de la structuration des données au sein de systèmes NoSQL orientés documents. La flexibilité et le grand nombre de possibilités offertes par ces systèmes sont aussi bien des atouts que des aspects difficiles à maîtriser. Nous considérons une structure de données ou "schéma" de référence pour la base de documents, comme un produit. La section 4.1 introduit l'approche. Les sections suivantes détaillent les étapes de la ligne de produits basée sur les modèles de caractéristiques qui permettent d'obtenir a partir d'un modèle UML, des alternatives de structuration orientées document.

4.1. Modèles de caractéristiques et structuration orientée document

Nous utilisons un modèle de caractéristiques qui permet de représenter les différences de structuration et de contrôler le nombre de possibilités pour qu'il reste raisonnable. Les différences de structuration seront traitées comme des variantes du produit final, les schémas. Ces variantes seront guidées par des directives de structuration orientées documents telles que l'imbrication, le référencement, la profondeur et la duplication. Les configurations valides résultantes d'une telle modélisation seront donc un ensemble d'alternatives à analyser.

Nous proposons de partir d'un modèle UML donné par l'utilisateur et de créer un modèle de caractéristiques permettant de proposer des alternatives de structuration orienté documents. La Figure 5 illustre le cas de base. Les éléments du modèle UML seront notés comme $mU = (E,R)$ où :

- $E = \{e_1,...,e_n\}$ est l'ensemble des classes
- $R = \{r_1,...,r_n\}$ est l'ensemble d'associations.
- $R(e_i) = \{r_1,...,r_n\}$ est l'ensemble d'associations de la classe e_i
- $E(r_n) = \{e_i,e_j\}$ sont les classes reliées par l'association r_n. $e_i \neq e_j$
- $card(r_n,e_i)$ et $rol(r_n,e_i)$ sont la cardinalité et le rôle de l'association r_n par rapport à la classe e_i.
- $A(e_i) = \{a_1,...,a_n\}$ sont les attributs de la classe e_i. $a_i = \{a_i.name : a_i.type\}$
- $te_i = \{A(e_i),a_{id}\}$ est le type de la classe e_i. Il est composé de l'ensemble d'attributs de la classe et d'un identifiant par défaut [1].

Dans cet article nous nous limitons aux associations binaires sans attributs.

1. Ce type est introduit pour faciliter l'explication de nos propositions.

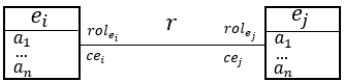


Figure 5. Cas de base : deux classes et une association

4.2. Modélisation de variations entre alternatives

Nous modélisons les variations des alternatives orientées documents avec un modèle de caractéristiques fms permettant d'exprimer les options d'imbrication et référencement. Les principaux choix de modélisation sont les suivants :

1. Une classe e_i peut déclencher la création d'une collection de documents de type te_i.
2. Une association r peut être matérialisée par l'imbrication ou par le référencement de e_i dans e_j ou vice-versa.
3. Une association r peut être matérialisée par une collection-lien.

Pour introduire le modèle de caractéristiques nous allons considérer d'abord le cas simple de deux classes et une association, tel qu'illustré sur la Figure 5. La modélisation de chacune des classes est abordée comme suit.

Modélisation d'une classe
Pour modéliser la variabilité des alternatives de structuration d'une collection pour une classe e_i, nous proposons le modèle de caractéristiques fm_{col}, introduit par la zone grise de la Figure 6a. Ses caractéristiques permettent de traiter l'information propre une classe e_i et une association r :

– La caractéristique "racine", nommée $cole_i$, définit la collection pour la classe e_i.

– Le type des documents de cette collection (au premier niveau) est défini comme une caractéristique fille obligatoire, nommée te_i. Ce type regroupe les attributs de e_i et l'identifiant a_{id}.

La modélisation des associations de la classe introduisent plusieurs variantes. Parmi elles, l'extension du type te_i par ajout d'attributs au niveau 0. Cette possibilité est introduite par la caractéristique optionnelle $te_i\_l_0\_ext$, fille de la caractéristique te_i. Les alternatives de matérialisation de l'association r sont définies par un groupe XOR dépendant de la caractéristique r_role_j. Ce groupe a deux alternatives qui modélisent soit l'imbrication, soit le référencement de documents e_j:

1. La caractéristique $te_i\, EMB^{ce_j}\, te_j$ indique l'*imbrication* dans te_i d'un ou de plusieurs documents du type te_j. Une cardinalité ce_j "plusieurs" indique l'imbrication d'un tableau de documents, une cardinalité 1 indique l'imbrication d'un seul document.

2. La caractéristique $te_i\, REF^{ce_j}\, te_j$ indique le *référencement* depuis un type te_i étendu, d'un ou plusieurs documents du type te_j selon la cardinalité ce_j. Une référence ou un tableau de références. Les références se font en utilisant l'identifiant a_{id}.

Modélisation de deux classes et une association
Basé sur la modélisation de caractéristiques d'une collection, fm_{col}, nous modélisons maintenant la variabilité des alternatives de deux classes et une association. Il s'agit du modèle de caractéristiques fm_{asso} (cf. Figure 6) comme suit :

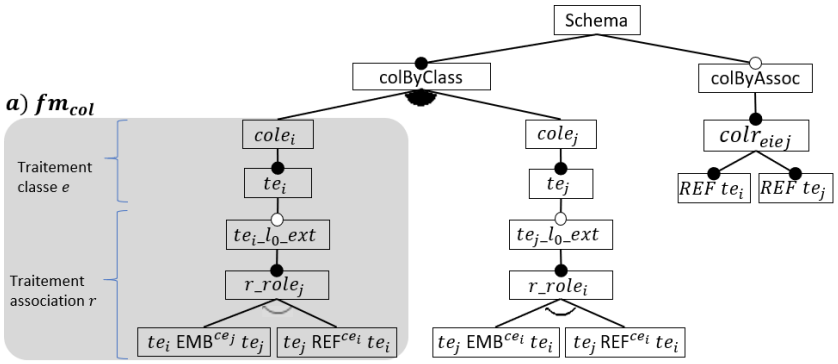


Figure 6. Modèle de caractéristiques fm_{asso} : deux classes et une association

– La caractéristique racine, nommée *Schema*, a deux caractéristiques filles, une obligatoire, nommée *colByClass*, et une optionnelle, nommée *colByAssoc*.

– *colByClass* détermine avec un groupe OR, la ou les collections qui peuvent exister en fonction des classes du modèle UML. Chaque caractéristique fille correspondra à un fm_{col} par classe e_k.

– *colByAssoc* définit une *collection-lien* par association avec une caractéristique nommée $colr_{e_ie_j}$. Celle-la référence des documents de type te_i et te_j.

Modélisation de plusieurs classes et associations
Pour traiter un modèle UML avec plusieurs classes et associations nous proposons l'algorithme *AMISS*, introduit dans la section 5. *AMISS* utilise la modélisation du cas simple (deux classes et une association) que nous venons de présenter. Un parcours du modèle UML crée de manière incrémentale le modèle de caractéristiques complet, nommée fm_s. Ce modèle représente toutes les alternatives de structuration proposées pour une base orientée documents. Le modèle fm_s inclut une collection par classe et par association (pour les éventuelles colletions-lien). Pour chaque collection de classe, le type des éléments inclut les attributs propres à la classe et des possibles attributs supplémentaires pour représenter les associations de la classe.

4.3. Définition des structurations valides

Le modèle de caractéristiques introduit précédemment génère plusieurs alternatives de structuration. Nous avons établi un ensemble de contraintes pour assurer que, par construction, toutes les configurations fournies par le modèle de caractéristiques sont valides. Ces contraintes couvrent les garanties de structuration de la section 2 et servent à diminuer l'explosion combinatoire des possibilités ou des alternatives de configuration. Les contraintes $\mathscr{D}$ sont classées en cinq groupes :

Les contraintes d'isolement $\mathscr{D}^I$, empêchent l'isolement d'une collection. Si une collection de type te_i, au premier niveau, n'a pas d'extension pour son association r_n vers te_j, alors cette association doit être considérée soit depuis une collection *lien* soit depuis un niveau imbriqué appartenant à une autre collection :

$$\mathscr{D}^I = \{\, te_i \wedge \neg te_i\_l_0\_ext \implies cole_ie_j \vee r\_role_j \;,\; te_j \wedge \neg te_j\_l_0\_ext \implies cole_je_i \vee r\_role_i \,\}$$

Les contraintes d'existence $\mathscr{D}^E$ garantissent l'existence d'une collection référencée. Si un type te_i est référencé, alors une collection du même type doit exister.

$$\mathscr{D}^E = \{\, te_i REF^{ce_j} te_j \implies cole_j \;,\; te_j REF^{ce_j} te_i \implies cole_i \,\}$$

Les contraintes de boucles imbriquées $\mathscr{D}^{BI}$ empêchent les collections d'être référencées entre elles. Si un te_i réfère le type te_j, alors te_j ne peut pas imbriquer te_i.

$$\mathscr{D}^{BI} = \{\, te_i REF^{ce_j} te_j \implies \neg te_j EMB^{ce_i} te_i \;,\; te_j REF^{ce_i} te_i \implies \neg te_i EMB^{ce_j} te_j \,\}$$

Les contraintes de références bouclées $\mathscr{D}^{RB}$ empêchent une collection qui imbrique des documents de type te_i, d'être référencée par une autre collection du même type. Si un te_i réfère le type te_j, alors te_j ne peut pas référencer te_i.

$$\mathscr{D}^{BR} = \{\, te_i REF^{ce_j} te_j \implies \neg te_j REF^{ce_i} te_i \,\}$$

Les contraintes de *liens* d'association $\mathscr{D}^L$ garantissent qu'une *collection-lien* concernant une association r_n ne réfère que des collections dont le type n'est pas étendu par le deuxième type défini par r_n. Si le lien $cole_ie_j$ existe, on peut trouver une collection du type te_i non étendue par te_j et une collection du type te_j non étendue par te_i.

$$\mathscr{D}^L = \{\, cole_ie_j \implies (cole_i \wedge \neg r\_role_j) \wedge (cole_j \wedge \neg r\_role_i) \,\}$$

L'ensemble des contraintes du modèle de caractéristiques fm_{asso} sera donc formé par les contraintes d'isolement, d'existence, de boucles imbriquées, de références bouclées et de lien $\mathscr{D}(fm_{asso}) = \{\mathscr{D}^I, \mathscr{D}^E, \mathscr{D}^{BI}, \mathscr{D}^{BR}, \mathscr{D}^L\}$.

4.4. Configuration des alternatives

Compte tenu des contraintes $\mathscr{D}(fm_{asso})$, le modèle de caractéristiques fm_{asso} fournit huit configurations pour un modèle UML avec deux classes et une association [2]. Ces configurations correspondent aux alternatives de structuration introduites dans la Figure 2. Une configuration contient les noms des caractéristiques choisies. Voici la sémantique $\|fm_{asso}\|$ (configurations) du modèle fm_{asso} :

$$\|fm_{asso}\| = \{C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8\}$$

$C_1 = \{Schema, colByClass, \mathbf{cole_i}, te_i, te_i\_l_0\_ext, r\_role_j, te_iEMB^{ce_j}te_j\}$

$C_2 = \{Schema, colByClass, \mathbf{cole_i}, te_i, te_i\_l_0\_ext, r\_role_j, te_iEMB^{ce_j}te_j, \mathbf{cole_j}, te_j\}$

$C_3 = \{Schema, colByClass, \mathbf{cole_i}, te_i, te_i\_l_0\_ext, r\_role_j, te_iEMB^{ce_j}te_j,$
$\mathbf{cole_j}, te_j, te_j\_l_0\_ext, r\_role_i, te_jEMB^{ce_i}te_i\}$

$C_4 = \{Schema, colByClass, \mathbf{cole_j}, te_j, te_j\_l_0\_ext, r\_role_i, te_jEMB^{ce_i}te_i\}$

$C_5 = \{Schema, colByClass, \mathbf{cole_i}, te_i, \mathbf{cole_j}, te_j, te_j\_l_0\_ext, r\_role_i, te_jEMB^{ce_i}te_i\}$

$C_6 = \{Schema, colByClass, \mathbf{cole_i}, te_i, te_i\_l_0\_ext, r\_role_j, te_iREF^{ce_j}te_j, \mathbf{cole_j}, te_j\}$

$C_7 = \{Schema, colByClass, \mathbf{cole_i}, te_i, \mathbf{cole_j}, te_j, te_j\_l_0\_ext, r\_role_i, te_jREF^{ce_i}te_i\}$

$C_8 = \{Schema, colByClass, \mathbf{cole_i}, te_i, \mathbf{cole_j}, te_j, colByAssoc, \mathbf{colr_{e_ie_j}}\}$

2. Nous avons utilisé le *SAT solver* S.P.L.O.T. pour calculer le nombre de configurations valides. Cf. http://www.splot-research.org/ et http://52.32.1.180 8080/SPLOT/models/model_20190411_1457420991.xml

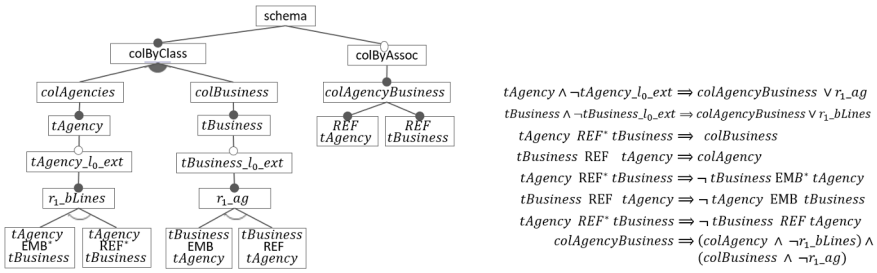


Figure 7. Modèle fm_{asso} correspondant pour le modèle UML de la Figure 1

Les configurations C_1 à C_5 correspondent aux combinaisons avec les choix d'imbrication de documents, alors que C_6 à C_8 correspondent aux choix de référencement. La configuration C_2, par exemple, contient les caractéristiques $cole_i$ et $te_i EMB^{ce_j} te_j$ indiquant l'existence d'une collection $cole_i$ avec imbrication de documents de type te_j. La caractéristique $cole_j$ indique la présence de la collection mais sans étendre son type, en l'absence de caractéristiques du genre EMB ou REF.

Reconsidérons le modèle UML de la Figure 1 avec les classes `Agency` et `Business` et l'association r_1 avec respectivement une cardinalité 1..* et des rôles `ag` et `bLines`. Le modèle de caractéristiques correspondant est introduit dans la Figure 7.

4.5. Dérivation d'une alternative de structuration

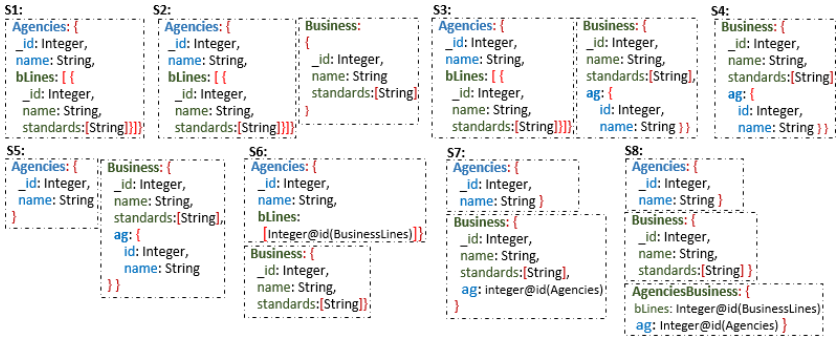


Figure 8. Produits du fm_{asso} de la Figure 7 sous forme d'AJSchema

Afin d'analyser les alternatives de structuration correspondant aux configurations fournies par le modèle de caractéristiques, chaque configuration est dérivée sous forme d'AJSchema. La Figure 8 montre les AJSchemas produits par le modèle de caractéristiques de la Figure 7[3]. Le format AJSchema est assez lisible et met en évidence les types des données. Pour enrichir l'analyse, il est possible d'évaluer automatiquement des métriques structurelles (Gómez *et al.*, 2018b) permettant de quantifier la complexité de la structure pour soit la privilégier, soit l'écarter.

3. Formalisation des structures illustrées par la Figure 2.

Algorithme 1 - AMISS : Algorithme de modélisation de schémas semi-structurés
Input: Modèle UML m, classes avec associations binaires ER
Output: Modèle de caractéristiques fm_s

```
1: fm_s ← null
2: e_i ← random(ER)
3: fm_s ← treatAssociations(fm_s, e_i)
4: fm_s ← treatIsolatedTypes(fm_s, E − ER)

5: function TREATASSOCIATIONS(fm_s, e_i) : fm_s
6:     foreach r_n ∈ R(e_i) do
7:         e_j ← getTarget(r_n, e_i)
8:         fm_asso ← generateFMassociation(e_i, e_j, r_n)
9:         if fm_s is null then
10:            fm_s ← fm_asso
11:        else
12:            fm_s ← FUSIONASSOCIATION(fm_s, fm_asso, r_n, e_i, e_j)
13:        end if
14:        fm_s ← TREATASSOCIATIONS(fm_s, e_j)
15:    end foreach
16:    return fm_s
17: end function
```

5. *AMISS* : Algorithme de ModélIsation de Schémas Semi-structurés

Nous avons vu la modélisation, fm_{asso}, pour deux classes et une association. Nous généralisons maintenant la proposition afin de créer un modèle de caractéristiques, fm_s, pour toutes les classes et associations d'un modèle UML. Pour cela, nous proposons l'*A*lgorithme de *M*odél*I*sation de *S*chémas *S*emi-structurés, *AMISS* (cf. Algorithme 1).

AMISS initie la création du modèle de caractéristiques en partant d'une des classes ayant des associations [4]. Une de ses associations est traitée et donne lieu à un modèle de caractéristiques fm_{asso} (Alg. 1, ligne 8). Ensuite, AMISS parcourt en profondeur les autres classes et associations du modèle UML. Chaque association entraîne la création d'un modèle de caractéristiques fm_{asso} qui est ensuite intégré au modèle complet fm_s (Alg. 1, ligne 12). Cette intégration est faite par fusion des modèles de caractéristiques.

L'algorithme FusionAssociation (cf. Algorithme 2) modifie fm_s pour intégrer la modélisation (représentée dans fm_{asso}) d'une nouvelle association r_n avec ses classes e_i et e_j. La Figure 9 illustre les actions principales (numérotées avec des étoiles) qui enrichissent fm_s.

1. Intégrer la modélisation de r_n dans la collection $cole_i$. Pour cela, ajouter la branche r_role_j du fm_{asso} aux caractéristiques présentes dans fm_s qui imbriquent des documents du type te_i (cf. Lignes 1-9).
2. Ajouter la modélisation de la nouvelle collection créee pour e_j. Le modèle fm_{col} correspondant a la branche $cole_j$ du fm_{asso} est ajouté au groupe OR de la caractéristique *ColByClass* de fm_s (cf. Lignes 10-11).
3. Compléter le type te_i imbriqué dans la nouvelle branche $cole_j$ de fm_s. Pour cela, lui ajouter la modélisation déjà existante des associations autres que r_n. Ces associations

4. Le modèle obtenu est indépendant du choix de la première classe à traiter.

Algorithme 2 - FusionAssociation $(fm_s, fm_{asso}, r_n, e_i, e_j)$: fm_s

▷ Compléter les types te_i à niveau 0 ou imbriqués
▷ Extraire branche r_role_j et contraintes

1: $fm_{branch} \leftarrow$ extractExtension(fm_{asso}, e_i, r_n)
2: $fm_s \leftarrow$ insertBranchCommonClass(fm_s, fm_{branch}, e_i)
3: **foreach** $f \in \mathcal{F}(fm_s) \wedge f = \bigotimes emb^{ce_i} te_i$ **do**
4: **if** f *is leaf* **then**
5: $fm_s \leftarrow$ addFeatureExtension$(fm_s, f, e_i, cse(e_i))$
6: **end if**
7: $fm_{branchV} \leftarrow$ createBranchVersion$(fm_{branch}, e_i, e_j, csr(r_n, e_j))$
8: $fm_s \leftarrow$ embedBranch$(fm_s, fm_{branchV}, f.child)$
9: **end foreach**

▷ Ajouter nouvelle collection $cole_j$

10: $fm_{col} \leftarrow$ extractFMcol(fm_r, e_j, r_n)
11: $fm_s \leftarrow$ insertFMcolNewClass(fm_s, fm_{col}, e_j)

▷ Compléter type te_i imbriqué de $cole_j$

12: $fm_s \leftarrow$ addFeatureExtension$(fm_s, te_j emb^{ce_i} te_i, e_i, cse(e_j))$
13: **foreach** fm_{branch} $filsOf$ $(te_i\_l_0\_ext) \in fm_s \bigwedge fm_{branch}.root \neq r\_role_j$ **do**
14: $fm_{branchV} \leftarrow$ createBranchVersion$(fm_{branch}, e_i, e_j, csr(r_n, e_i))$
15: $fm_s \leftarrow$ embedBranch$(fm_s, fm_{branchV}, e_j, cse_t)$
16: **end foreach**

▷ Ajouter *collection-lien* pour r_n

17: $fm_s \leftarrow$ insertBranchRelation$(fm_s, fm_{asso}, r_n, e_i, e_j)$

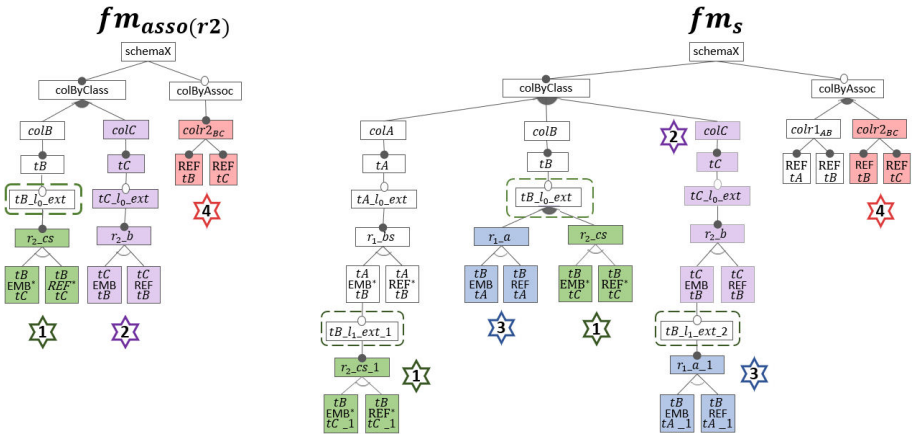


Figure 9. Exemple de fusion: sur le côté gauche modèle fm_{asso} à intégrer, sur la droite modèle fm_s modifié pour intégrer fm_{asso}

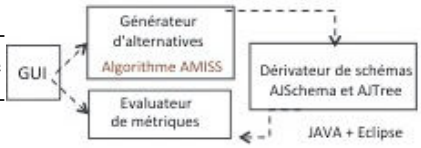
se trouvent dans le type te_i de $cole_i$ dans fm_s (cf. Lignes 12-16).
4. Ajouter la *collection-lien* pour r_n. Celle-ci est représentée par la caractéristique $colasso_{e_i e_j}$ de fm_{asso} et doit être insérée dans l'arbre de la caractéristique *colByAssoc* de fm_s (cf. Ligne 17).

Le modèle fm_s obtenu représente un ensemble de configurations valides où chaque configuration correspond à une alternative de structuration. AMISS crée un arbre de caractéristiques qui peut donner un très grand nombre de configurations valides. Dans le cas de trois classes et deux associations cela correspond à 106 configurations valides. Le nombre de configurations valides croît de manière exponentielle avec le nombre d'associations. Notre préconisation est de restreindre ce nombre en introduisant des contraintes tel qu'illustré dans la section 4.3. Ces contraintes peuvent représenter des préférences de structuration données par l'utilisateur ou être de bonnes pratiques

connues dans le contexte où la base de documents sera utilisée. Par exemple, limiter le nombre d'imbrications de documents, le nombre de références à une collection précise ou le nombre d'attributs complexes par type de document.

6. Mise en œuvre de *ScorusTool*

Le prototype *ScorusTool* implémente la génération de "schémas" mais aussi l'évaluation de métriques structurelles. Ses principaux composants sont illustrés ci-contre.



Le « *Générateur d'alternatives* » met en œuvre les propositions de cet article. Il récupère un modèle UML fourni par l'utilisateur et crée un modèle de caractéristiques afin de fournir un ensemble d'alternatives de structuration orientées document. Le framework *EMF* (Steinberg *et al.*, 2008 ; Eclipse, s. d.) ainsi que la spécification du méta-modèle d'UML (OMG, s. d.) ont été utilisés. Le langage *FAMILIAR* (Acher *et al.*, 2013) a servi pour la création du modèle de caractéristiques. Il permet de manipuler et de raisonner sur les variantes d'un modèle de caractéristiques. L'algorithme *AMISS* a été implémenté à l'aide de *FAMILIAR*. Chacune des configurations, correspondant aux alternatives de structuration, est fournie sous la forme d'un ensemble de noms des caractéristiques sélectionnées (cf. Section 4.4).

Le « *Dérivateur de schémas* » traduit une configuration du modèle fm_s, correspondant à une alternative de structuration de données sous la forme d'AJSchéma (illustré sur la Figure 8) et d'une arborescence, AJTree. Ce composant utilise notamment des bibliothèques de manipulation de graphes en Java. L'AJTree est utilisé pour l'évaluation automatique de métriques structurelles effectué par « *l'évaluateur de métriques* ». Ces métriques ont été présentées dans (Gómez *et al.*, 2018b).

Une interface graphique est proposée à l'utilisateur. Celle-ci montre les configurations fournies par le modèle de caractéristiques fm_s correspondant aux alternatives de structuration. Pour chaque alternative, l'outil montre l'AJTree et l'AJSchéma ainsi que les valeurs des métriques structurelles.

7. Travaux connexes

L'utilisation des modèles de caractéristiques pour générer des alternatives de structuration de données orientées document, s'est avéré pertinente. Cette approche est, à notre connaissance, originale pour la conception de bases de données. Elle permet d'explorer et de gérer un grand nombre d'alternatives de structuration et ouvre des possibilités intéressantes pour la prise en compte de contraintes.

Certains travaux s'intéressent à la modélisation de données pour les bases NoSQL en utilisant d'autres approches. La motivation de la plupart de ces travaux est le choix, presque ad-hoc, d'une structure de données favorable aux performances d'exécution d'un ensemble de requêtes donné. (Zhao *et al.*, 2014) présente un algorithme pour la création systématique d'une base de documents à partir d'un modèle entité-relation. Cet algorithme propose une dé-normalisation de ce modèle avec pré-calcul des join-

tures naturelles par imbrication de documents. La solution engendre en général de la redondance des données.

Des travaux tels que (Mior *et al*., 2017), (Lombardo *et al*., 2012) et (Vajk *et al*., 2013) s'intéressent aux alternatives de "modélisation" des données dans Cassandra (modèle "big table"). Les objectifs des études portent sur le stockage et les performances des requêtes.(Lombardo *et al*., 2012) propose la création de plusieurs versions des données avec des structures différentes selon les requêtes pré-calculées.

Les approches présentées dans (Abdelhedi *et al*., 2017 ; Atzeni *et al*., 2016) sont plus riches que les autres et ciblent plusieurs systèmes avec des modèles de données différents. Ces travaux s'appuient sur un méta-modèle pour représenter les caractéristiques des modèles de données NoSQL et pouvoir créer à partir de la même information, des alternatives de structuration selon le système cible. L'approche de (Abdelhedi *et al*., 2017) est dirigée par les modèles (Kleppe *et al*., 2003). Ils proposent de transformer le modèle UML dans un modèle logique générique contenant les concepts communs des trois modèles (Cassandra, Neo4J et MongoDB). Ensuite, une phase de transformation utilise le modèle générique et crée des alternatives de structuration pour chaque système NoSQL. Ils proposent des règles de transformation d'une association selon chaque système. Pour MongoDB, 5 solutions sont proposées. La manière d'interpréter la présence de plusieurs associations dans le modèle n'est pas introduite.

8. Conclusion et perspectives

La flexibilité de la structuration des données offerte par les systèmes NoSQL orientés documents, comme MongoDB, permet de nombreuses options de modélisation, chacune avec ses avantages et ses inconvénients en fonction de plusieurs facteurs.

Tenant compte des enjeux et de la difficulté du choix, nous proposons à l'utilisateur d'étudier plusieurs alternatives semi-structurées à l'aide d'un générateur qui travaille sur un modèle UML des données à gérer. Nous adoptons une approche de lignes de produits logiciel qui permet de créer automatiquement une modélisation de la variabilité des structures susceptibles d'être des "schémas" de référence pour la base de documents. Les algorithmes que nous proposons fonctionnent avec un modèle de caractéristiques où (1) les variantes reflètent la flexibilité du système de types des bases de documents et (2) les contraintes représentent un savoir-faire de modélisation pour obtenir des variantes qui ont du sens. L'explosion du nombre de variantes est ainsi contrôlé mais l'ensemble de contraintes peut évoluer pour intégrer de nouvelles contraintes appropriées au contexte d'utilisation des bases de documents à construire.

Les propositions de cet article sont implantées par le prototype *ScorusTool* qui inclut aussi l'évaluation de métriques structurelles sur les alternatives de structuration. Les perspectives de recherche portent d'abord sur des expérimentations puis sur des extensions pour faciliter l'ajout des contraintes d'un problème particulier afin de réduire le nombre d'alternatives à évaluer.

Remerciements: Nous remercions G. Vega, J. Chavarriaga, JP. Giraudin et les relecteurs anonymes pour leur retours précieux.

Bibliographie

Abdelhedi F., Brahim A. A., Atigui F., Zurfluh G. (2017). MDA-Based approach for NoSQL databases modelling. In *International Conference on Big Data Analytics and Knowledge Discovery.* Springer.

Acher M., Collet P., Lahire P., France R. B. (2013). FAMILIAR: A domain-specific language for large scale management of feature models. *Science of Computer Programming (SCP) , Journal*, vol. 78, nº 6, p. 657-681.

Atzeni P., Bugiotti F., Cabibbo L., Torlone R. (2016). Data modeling in the NoSQL world. *Computer Standards & Interfaces Journal*.

Eclipse. (s. d.). *Eclipse Modeling Framework Project (EMF).* www.eclipse.org/modeling/emf/. (Accessed: 2018-09-21)

Gomez P. (2018). *Analyse et évaluation de structures orientées document*. Thèse. Université Grenoble Alpes. https://www.theses.fr/2018GREAM076.

Gómez P., Casallas R., Roncancio C. (2016). Data schema does matter, even in NoSQL systems!. In *Research Challenges in Information Science (RCIS) Tenth Intern. Conference on.* IEEE.

Gómez P., Roncancio C., Casallas R. (2018a). Métriques structurelles pour l'analyse de bases orientées documents. In *Actes du xxxvième Congrès INFORSID.*

Gómez P., Roncancio C., Casallas R. (2018b). Towards quality analysis for document oriented bases. In *International Conference on Conceptual Modeling (ER).* Springer.

Kang K. C., Cohen S. G., Hess J. A., Novak W. E., Peterson A. S. (1990). *Feature-oriented domain analysis (FODA) feasibility study*. Rapport technique. Carnegie-Mellon Univ Pittsburgh Pa Software Engineering Inst.

Kang K. C., Lee J., Donohoe P. (2002). Feature-oriented product line engineering. *IEEE Software Magazine*, vol. 19, nº 4, p. 58–65.

Kleppe A. G., Warmer J., Bast W., Explained M. (2003). *The model driven architecture: practice and promise.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA.

Lombardo S., Nitto E. D., Ardagna D. (2012). Issues in handling complex data structures with NoSQL databases. In *14th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC).*

Mior M. J., Salem K., Aboulnaga A., Liu R. (2017). Nose: Schema design for NoSQL applications. *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, nº 10, p. 2275–2289.

OMG. (s. d.). *Unified Modeling Language (UML) 2.0.* http://www.omg.org/spec/UML/2.0/. (Accessed: 2018-09-21)

Steinberg D., Budinsky F., Merks E., Paternostro M. (2008). *EMF: eclipse modeling framework*. Livre. Pearson Education.

Vajk T., Feher P., Fekete K., Charaf H. (2013). Denormalizing data into schema-free databases. In *Cognitive Infocommunications (CogInfoCom), 4th International Conference on.* IEEE.

Zhao G., Lin Q., Li L., Li Z. (2014, 11). Schema conversion model of SQL database to NoSQL. In *P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2014 Ninth International Conference on.*

Discovery of Genuine Functional Dependencies from Relational Data with Missing Values

Laure Berti-Équille [1,2], **Hazar Harmouch** [3], **Felix Naumann** [3], **Noël Novelli** [1], **Saravanan Thirumuruganathan** [4]

1. *Aix Marseille Université, Univ. de Toulon, CNRS, LIS, DIAMS, Marseille, France*
   laure.berti-equille@univ-amu.fr,noel.novelli@lis-lab.fr
2. *ESPACE-DEV/IRD, UMR 228, IRD/UM/UG/UR, Montpellier, France*

3. *Hasso Plattner Institute, University of Potsdam, Germany*
   hazar.harmouch@hpi.de,felix.naumann@hpi.de
4. *QCRI, HBKU, Doha, Qatar*
   sthirumuruganathan@hbku.edu.qa

ABSTRACT. *This article is an extended abstract of our work published at VLDB'2018. The full paper is available at www.vldb.org/pvldb/vol11/p880-berti-equille.pdf.*
Functional dependencies (FDs) play an important role in maintaining data quality in relational databases. They can be used to enforce data consistency and guide data repairs. In this work, we investigate the problem of missing values and its impact on FD discovery. When using existing FD discovery algorithms, some genuine FDs could not be detected precisely due to missing values and some non-genuine FDs can be discovered even though they are caused by missing values depending on the considered semantics for NULL values. We define the notion of genuineness of FDs and propose algorithms to compute the FD genuineness score. This can be used to identify genuine FDs *among the set of all valid dependencies that hold on the data. We evaluate the quality of our method over various real-world and semi-synthetic datasets with extensive experiments. The results show that our method performs well for relatively large FD sets and is able to accurately capture genuine FDs.*

KEYWORDS: *Functional dependencies, missing values, scoring.*

---

1. Context and motivations

Functional dependencies (FDs) are one of the most important types of integrity constraints and have been extensively studied by the DB research community. FDs

have a number of applications, such as maintaining data quality in databases, capturing schema semantics, schema normalization, data integration, repairing of data inconsistencies, and data cleaning. An FD $X \rightarrow A$ states that the tuples of attribute set X uniquely determine the value of attribute (set) A. Traditional FDs are typically defined for correct and complete data and there are many efficient algorithms to discover FDs from a given clean dataset. However, many real-world datasets are neither correct nor complete. Traditional FDs often have trouble with incomplete data, such as NULL values, that routinely exist in massive datasets with well-known data error rates that may vary from 20% up to 80%.

2. Genuine FDs discovery

Despite the importance of this problem, very few work has focused on the critical aspects of FD discovery over incomplete data. In our work, we consider three semantics of missing values: (i) all tuples with at least one NULL value are ignored in computing FDs; for each attribute, we substitute all NULL values either by (ii) the same value (all NULL values are considered equal) or (iii) by distinct values (all NULLs are considered distinct). Then, for each NULL semantics, we study the impact on FD discovery. We formally and experimentally show the phenomenon caused by missing values over FD discovery and we formalize the definitions of *genuine*, *ghost*, and *fake* FDs. Furthermore, we study their impact under various NULL semantics and imputation strategies.

Intuitively, given a clean dataset r and a corresponding dirty dataset r' polluted by injecting missing values in r: A *same FD* is a valid FD in r and also in r'. A *ghost FD* is a valid FD in r becomes invalid in r' while a *fake FD* is an invalid FD in r but is valid in r'. A *genuine FD* is a valid exact FD in r and in r'. To estimate FD genuineness score of FDs in a dirty relation, we propose (i) a probabilistic approach using a given imputation technique for estimating the score and we provide an efficient method for enumerating and pruning irrelevant possible worlds, (ii) we propose efficient algorithms to approximate the genuineness score of discovered FDs: the first one is based on possible worlds using *Monte Carlo* sampling and the second methods are based on probabilities per value and per tuple, using respectively the likelihood that the FD $X \rightarrow A$ which holds for the value $V_X \in Dom(X)$ can identify the value V_A. We estimate genuineness with *PerValue* score as the normalization of the sum of *PerValue* over the number of distinct values in X, and with *PerTuple* score as the normalization of the sum of $|V_X, V_A|$ over the number of distinct tuples. We performed extensive experiments of our methods on real-world (Sensors dataset) and semi-synthetic datasets (Abalone, Computer, Glass and Iris datasets) artificially polluted in a controlled experiments and showed the effectiveness and efficiency of our approach.

Acknowledgements

This work is funded by the French National Research Agency, project QualiHealth, ANR-18-CE23-0002 .

Une approche à base d'heuristiques pour la génération des requêtes spatio-temporelles à partir des questions en langage naturel

Ghada Landoulsi, Khaoula Mahmoudi, Sami Faiz

**Laboratoire de Télédétection et Systèmes d'Information à Références Spatiales,*
Ecole Nationale d'Ingénieurs de Tunis
BP 37 le Belvédère 1002 Tunis, TUNISIE

ghadalandoulsi@yahoo.fr, kamahmoudi@yahoo.fr, sami.faiz@insat.rnu

RESUME. De nos jours, les bases de données géographiques sont devenues largement omniprésentes et ont le potentiel d'assister les décideurs s'occupant des données spatiales, temporelles ou spatio-temporelles. Elles sont souvent amenées à être exploitées par des non-experts qui n'ayant pas une grande compétence et ne maîtrisant pas rigoureusement l'utilisation des systèmes d'information géographiques. Dans ce contexte, les systèmes de question réponse se sont développés pour faciliter l'accès au grand public en lui offrant la possibilité d'utiliser le langage naturel pour exprimer ses besoins informationnels. Dans ce présent article, nous proposons une approche basée sur deux phases: La première consiste à assurer le prétraitement de la question de l'utilisateur en langage naturel et la deuxième est dédiée à assurer la traduction de cette question en un format structuré à savoir une requête SQL spatio-temporelle.

ABSTRACT. Nowadays, geographic databases have become largely ubiquitous and have the potential to assist decision-makers dealing with spatial, temporal or spatio-temporal data. These latter, are often exploited by non-experts who do not have a great deal of expertise and do not master the use of geographical information systems rigorously. In this context, question answering systems are developed to facilitate access to the general public by offering them the possibility of using natural language to express their information needs. In this paper, we propose a two staged approach: The first stage is dedicated to ensure the preprocessing of the natural language user's question and the second is devoted to ensure the translation of this question into a structured format namely spatio-temporal SQL query.

Mots-clés : Langage naturel, bases de données géographiques, requête spatio-temporelle.

KEYWORDS: Natural language, geographic database, Spatio-temporal query.

1. Introduction

De nos jours, les Bases de données géographiques (BDG) représentent une source d'information très importante pour tous les types d'utilisateurs, experts ou non experts, stockant une partie importante des informations géographiques (Gesbert, 2004). Dans ce cadre, il est intéressant d'une part qu'un utilisateur soit capable d'accéder à une BDG et exploiter les informations stockées dedans. D'autre part, les Systèmes d'informations Géographiques (SIG) restent des outils de spécialistes qui nécessitent des compétences spécialisées, ce qui engendre un problème d'accès à l'information géographique par le grand public.

Notre objectif dans ce présent article consiste à faciliter l'accès à ces bases de données en utilisant le langage naturel comme moyen de communication. L'utilité d'une telle fonction appelée par la suite les systèmes de question réponse est évidente. Ces systèmes sont dédiés à faciliter la recherche et à réduire les obstacles entre un utilisateur non-expert et une machine. En revanche, cette dernière est incapable d'interpréter une question en langage naturel. Pour ce faire, nous proposons dans ce papier une nouvelle approche basée sur des techniques de Traitement Automatique du Langage Naturel (TALN). Ces dernières, permettent d'assurer l'interprétation et le prétraitement des questions des utilisateurs et les traduire en un format structuré capable d'extraire les informations souhaitées à partir d'une BDG.

Dans le reste de cet article, la section 2 est dédiée à une revue de la littérature. La section 3 détaille les deux phases de l'approche proposée. La section 4, expose les résultats de simulation de notre approche. Une évaluation est présentée dans la section 5 pour la validation de la contribution présentée. Finalement, une conclusion fournit un récapitulatif du travail réalisé et quelques perspectives qui découlent de cette étude.

2. Etat de l'art

Pour accéder à une base de données il est nécessaire d'utiliser un langage formel sophistiqué qui est le plus souvent le langage de requête structuré SQL. Dans ce contexte, un utilisateur qui ne connait pas la structure d'une base et n'avait aucune idée sur la syntaxe utilisée pour chercher l'information représente un des premiers problèmes intéressants que soulève l'interrogation des bases de données en langage naturel. En fait, pour chercher une information, les utilisateurs n'utilisent pas le même vocabulaire existant dans une base de données ce qui entraîne un problème d'ambiguïté.

Chaudhari (2013) propose d'utiliser un dictionnaire de synonymes afin d'améliorer le vocabulaire qu'il peut utiliser pour exprimer ses besoins en informations. À chaque fois qu'il choisit d'utiliser une nouvelle base de données il doit tout d'abord faire des entrées manuelles au niveau du dictionnaire. Dans le même contexte, plusieurs autres recherches ont donné de l'importance à l'aspect sémantique en plus de la gestion de la synonymie. Ils ont proposé de résoudre les

problèmes liés à la restriction de vocabulaires des bases de données afin d'assurer la correspondance soit entre la question de l'utilisateur et la réponse (Giordani et Moschitti, 2012) soit entre la question de l'utilisateur et la requête SQL (Giordani et Moschitti, 2009). Concernant la génération des requêtes SQL, plusieurs méthodes et techniques ont été proposées dans la littérature. Certains sont basées sur des grammaires probabilistes (Deshpande et Devale, 2012) ou des grammaires avec des règles prédéfinies (Alexander et al., 2013). D'autres ont utilisé un système d'apprentissage permettant d'exploiter la structure d'une base de données pour générer des requêtes SQL classifiées selon leurs degrés de plausibilité grâce à un SVM-ranker (Giordani et Moschitti, 2012). Ces méthodes ne fonctionnent que sur des bases de données particulières traitant différents types de données telles que des données médicales (Abacha, 2012), des données géographiques (Chen, 2014), etc. Plus récemment, une approche à base d'esquisse est proposée par Yaghmazadeh et al. (2017). Une esquisse est caractérisée par une grammaire plus simple que le SQL et elle est déterminée en utilisant un système à base de règles, puis raffinée et réparée de manière itérative en se basant sur un ensemble de règles et d'heuristiques. Pour la génération d'une esquisse de requête, Yaghmazadeh et al. (2017) utilise l'apprentissage automatique et Word2Vec (Mikolov et al., 2013). En fait, elle représente le squelette d'une requête SQL avec ses différentes clauses, mais elle ne contient aucune information sur le schéma de la base de données utilisée. En s'inspirant des différentes méthodes proposées dans les travaux présentés dans la littérature, et afin d'accomplir le processus de génération des requêtes spatio-temporelles, nous proposons dans ce présent travail une approche à base d'heuristiques.

3. Approche proposée

Dans ce papier, nous proposons une approche à base de deux phases : La première consiste à assurer le prétraitement de la question de l'utilisateur en langage naturel et la deuxième est dédiées à assurer la traduction de cette question en un format structuré à savoir une requête SQL spatio-temporelle.

3.1. Prétraitement des questions des utilisateurs

Cette étape est caractérisée principalement par les trois étapes suivantes : étiquetage morphosyntaxique, génération de l'arbre de dépendance et la reconnaissance des entités et des relations géographiques.

3.1.1 Étiquetage morphosyntaxique

L'étiquetage morphosyntaxique consiste à attribuer la catégorie grammaticale (notée Cat_g) qui correspond à chaque mot de la question de l'utilisateur (noté QU) (Ferraro et al., 2013). La figure 1(1) présente le processus d'étiquetage morphosyntaxique de la question suivante : *« Determine the country where the wildfire disaster reaches the highest number of victims in 2014 ».* Une variété d'étiquettes morphosyntaxiques peut être utilisée pour désigner la catégorie

grammaticale des mots telles que : les adjectifs comparatifs (JJR), les noms propres singuliers (NNP), les adverbes comparatifs (RBR), etc.

3.1.2 Génération de l'arbre de dépendance

La génération de l'arbre de dépendance est dédiée à extraire les différentes relations lexicales qui existent entre chaque paire de mots en utilisant l'analyseur syntaxique de Marneffe et al, (2006). Ce dernier permet en fait de spécifier le lien syntaxique entre deux mots W_i et W_j. Dans une question (QU) il est possible d'avoir des mots composés dont la relation de dépendance (notée DRel) est « compound ». Dans ce cas, nous proposons un ensemble d'heuristiques permettant de combiner les noms composés et de les traiter en tant qu'un seul mot (figure 1(2)):

- Heuristique 1 : $\forall W_i, W_{i+1} \in QU\ (Cat_g(W_i) = WRB) \wedge (Cat_g(W_{i+1}) = JJ \vee RB) \wedge (DRel(W_{i+1}, W_i) = advmod) \Rightarrow Cat_g(W_i\_W_{i+1}) = RB$

Les mots composés tels que (how many, how longer, how far, etc.) doivent être considérés comme un seul mot. Dans ce cas, cette heuristique peut être interprétée comme suit : quelque soit un mot i (W_i) et un mot i+1 (W_{i+1}) appartenant à une question d'utilisateur (QU), Si W_i, un adverbe en WH (WRB) est suivi par un adjectif (JJ) ou par un adverbe (RB) (W_{i+1}) et que leur relation de dépendance notée DRel(W_{i+1}, W_i) représente un modificateur adverbial (advmod) alors on considère W_i et W_{i+1} en tant qu'un seul adverbe (RB).

- Heuristique 2 : $\forall W_i, W_{i+1} \in QU\ (Cat_g(W_i) = NN \vee NNS) \wedge (Cat_g(W_{i+1}) = NN \vee NNS) \wedge (DRel(W_{i+1}, W_i) = compound) \Rightarrow Cat_g(W_i\_W_{i+1}) = NN \vee NNS$

Heuristique 2 indique que si un nom (W_i) singulier (NN) ou pluriel (NNS) est suivi par un autre nom W_{i+1} et que leur relation de dépendance DRel(W_{i+1}, W_j) est compound, alors nous considérons W_i et W_{i+1} comme un seul mot dont la catégorie grammaticale est un nom (NN ou NNS).

- Heuristique 3 : $\forall W_i, W_{i+1}, W_{i+2} \in QU\ (Cat_g(W_i) = NN \vee NNS) \wedge (Cat_g(W_{i+1}) = IN) \wedge (Cat_g(W_{i+2}) = NN \vee NNS) \wedge (DRel(W_{i+2}, W_i) = nmod) \Rightarrow Cat_g(W_i\_W_{i+1}\_W_{i+2}) = NN \vee NNS$

La troisième heuristique suppose que Si W_i et W_{i+2} sont deux noms (NN ou NNS) reliés par une préposition ou une conjonction de subordination (IN) et que leur relation de dépendance DRel(W_{i+2}, W_i) est un modificateur nominal (nmod), alors W_i et W_{i+1} et W_{i+2} peuvent être considérés comme un seul mot.

- Heuristique 4 : $\forall W_i, W_{i+1} \in QU\ (Cat_g(W_i) = JJ) \wedge (Cat_g(W_{i+1}) = NN \vee NNS) \wedge (DRel(W_{i+1}, W_i) = adjmod) \Rightarrow Cat_g(W_i\_W_{i+1}) = NN \vee NNS$

Si Wi est un adjectif (JJ) suivi par un nom W_{i+1} et que leur relation de dépendance DRel(W_{i+1}, W_i) est un modificateur adjectival (adjmod) tel que (last year, previous day, etc) alors W_i et W_{i+1} peuvent être considérés comme un seul mot.

- Heuristique 5 : $\forall W_i, W_{i+1}, W_{i+2} \in QU\ (Cat_g(W_i) = JJ) \wedge (Cat_g(W_{i+1}) = CD) \wedge (Cat_g(W_{i+2}) = NN \vee NNS) \Rightarrow Cat_g(W_i\_W_{i+1}\_W_{i+2}) = NN \vee NNS$

Dans ce cas, W_i et W_{i+2} peuvent être reliés par un nombre cardinal (CD) W_{i+1} tel que (last three years) dans ce cas, W_i, W_{i+1} et W_{i+2} peuvent être traités comme un seul mot.

3.1.3 Reconnaissance des entités et des relations géographiques

En général, une question spatio-temporelle implique une recherche dans les dimensions spatiales et temporelles. Ce type de question est dédié à décrire des événements en identifiant leur localisation, leur date, leur durée, leur relation temporelle et leur relation spatio-temporelle. Dans la littérature, quatre types de requêtes spatio-temporelles sont mentionnés (Yuan et McIntosh, 2002) : Requête spatio-temporelle simple, requête spatio-temporelle d'intervalle, requête spatio-temporelle comportementale et requête spatio-temporelle relationnelle. Dans ce présent article, nous proposons de traiter les requêtes spatio-temporelles simples et les requêtes spatio-temporelles d'intervalles.

Le type de requêtes spatio-temporelles simples est destiné à chercher des informations sur la position d'un objet à un moment donné, le moment où un objet peut exister à une localisation particulière ou l'objet pouvant exister à un endroit et à un moment donné. En ce qui concerne le type de requêtes spatio-temporelles d'intervalles, il est dédié à chercher des informations sur les changements pouvant survenir dans une région au cours d'une période donnée. Dans ce cadre, les utilisateurs peuvent utiliser des expressions temporelles ou spatiales ambigües et imprécises pour décrire leurs besoins en informations. Pour identifier ces ambiguïtés, nous avons utilisé un outil permettant de reconnaitre les entités nommées (Manning et al., 2014). Dans le contexte des catastrophes naturelles, toutes les entités géographiques sont étiquetées avec la classe LOCATION, telle que les forêts, les terres, les pays, etc. Les types de catastrophes sont identifiés comme étant DISASTERS. Concernant les expressions temporelles, nous avons adopté un étiqueteur temporel basé sur des règles (Chang et Manning, 2012) supportant cinq types d'expressions temporelles : DATE, TIME, DURATION, SET et INTERVAL.

Dans les études antérieures, seules les relations spatiales (Worboys, 1992) ou temporelles (Allen, 1983) sont prises en compte. Les relations spatiales valides pour une période de temps particulière sont classées comme des relations spatio-temporelles (Raza, 2008). Ces dernières sont identifiées à l'aide d'un ensemble d'expressions régulières (Chang et Manning, 2014) capables de détecter chaque type de relation. De plus, si un utilisateur introduit une question de type WH, dans ce cas, des mots tels que (Where, when, how longer, etc) doivent être identifiés. Ces mots révèlent indirectement ce que l'utilisateur cherche. Par exemple, le mot « Where » peut être étiqueté avec la classe LOCATION et « How_longer » avec la classe DURATION. La figure 1(3) illustre un exemple de l'étape de reconnaissance des classes des entités et des relations géographiques notées Geo_{ERC}.

6

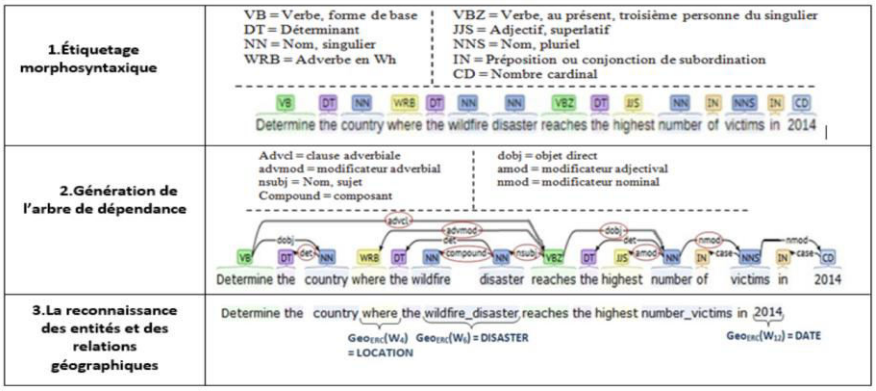


Figure 1. Les différentes étapes de la phase de prétraitement

3.2. Génération des requêtes spatio-temporelles

Pour parvenir à une traduction efficace des questions en requêtes spatio-temporelles, nous proposons d'utiliser une formulation qui conserve la syntaxe standard des requêtes SQL classiques, avec l'intégration de nouvelles fonctions et agrégats liés aux aspects spatiaux, temporels et spatio-temporels. Cette formulation présente la syntaxe suivante : *SELECT nom(s)_attributs(s) FROM nom(s)_table(s) WHERE condition(s) GROUP BY nom(s)_attribut(s) HAVING condition(s).*

3.2.1. Construction de l'instruction SELECT

Le processus de construction de l'instruction SELECT est réalisé en s'appuyant sur un ensemble d'heuristiques (Algorithme 1). Ces dernières sont basées sur les relations et les entités spatiales, temporelles et spatio-temporelles reconnues, la liste des étiquettes morphosyntaxiques et la liste des relations de dépendance.

Algorithme 1. Construction de l'instruction SELECT

---

1. **Entrée** *QU = {w1, w2, w3…,wn },* WHC // WHC = Classe du mot de question en WH, *QU = { w1, w2, w3…,wn } :* QU est constituée d'un ensemble de mots
2. **Sortie** $Sst_{att}L$ // $Sst_{att}L$ = liste des attributs de l'instruction Select
3. **Début**
4. $Sst_{att}L \leftarrow \varnothing$
5. **Pour tout** $W_{i \in \{1, 2, 3, n\}}$ **Faire**
   //Heuristique 1
6. **Si** ($Cat_g(W_i) \in$ {NN, NNS}) et ($DRel(W_j, W_i) \in$ {root,nsubj, det, dobj} **Alors** \\ NN = nom singulier, NNS = nom pluriel, root = racine, nsubj = nom sujet, det = déterminant, dobj = objet direct
7. Ajouter W_i à $Sst_{att}L$
8. **Fin Si**
   //Heuristique 2

9. **Si** ($Cat_g(W_i)$ ∈ {NN, NNS}) et ($Cat_g(W_j)$ ∈ {NN, NNS}) et ($DRel(W_j, W_i)$ ∈ {conj:and})) **Alors** // conj:and = Conjonction
10. Ajouter W_i et W_j à $Sst_{att}L$
11. **Fin Si**

//Heuristique 3

12. **Si** ($Cat_g(W_i)$ ∈ {NN, NNS}) et ($Cat_g(W_j)$ ∈ {NNP}) et ($DRel(W_j, W_i)$ ∈ {nmod, compound}) **Alors** \\ nmod = modificateur nominal, compound = composé
13. Ajouter W_i à $Sst_{att}L$
14. **Fin Si**

//Heuristique 4

15. **Si** ($Cat_g(W_i)$ ∈ {NN, NNS}) et ($Cat_g(W_j)$ ∈ {NN, NNS}) et ($DRel(W_j, W_i)$ ∈ {nmod}) **Alors**
16. Ajouter W_i et W_j à $Sst_{att}L$
17. **Fin Si**

//Heuristique 5

18. **Si** ($Cat_g(W_i)$ ∈ {WRB, WP}) **Alors** //WRB = adverbe WH, WP = pronom WH
19. Ajouter la classe de mot WH (WHC) du mot W_i à $Sst_{att}L$
20. **Fin Si**
21. **Fin pour**
22. **Fin**

3.2.2. Construction de la clause WHERE

Conjointement avec l'instruction SELECT il est indispensable d'utiliser la clause WHERE (notée Wc) pour filtrer les données. Pour ce faire, nous proposons un ensemble d'heuristiques qui sont déterminées en se référant aux classes d'entités et de relations géographiques (notées Geo_{ERC}) et aux catégories grammaticales (Cat_g).

Algorithme 2. Construction de la clause WHERE

1. **Entrée** $QU = \{w_1, w_2, w_3 \ldots, w_n\}$, Geo_{ERC} // Geo_{ERC} : la classe des entités et des relations géographiques
2. **Sortie** $Wc_{conditions}$ // liste des conditions de la clause WHERE
3. **Début**
4. $Wc_{conditions} \leftarrow \varnothing$
5. **Pour tout** $W_{i \in \{1, 2, 3, n\}}$ **Faire**

//Heuristique 1

6. **Si** ($Cat_g(W_i)$ ∈ {NNP, NNPS, CD}) **Alors** // NNP: nom propre singulier, NNPS: nom propre pluriel,CD: nombre cardinal
7. $Wc_{conditions} \leftarrow (Geo_{ERC}(W_i) = \text{`}W_i\text{'})$

//Heuristique 2

8. **Si** ($Cat_g(W_i)$ ∈ {NN, NNS}) suivi par ($Cat_g(W_{i+1})$ ∈ {VBZ}) suivi par ($Cat_g(W_{i+2})$ ∈ {NNP, NNPS}) **Alors** //NN : nom singulier, NNS : nom pluriel, VBZ : Verbe, au présent, troisième personne du singulier
9. $Wc_{conditions} \leftarrow (W_i = \text{`}W_{i+2}\text{'})$

//Heuristique 3

10. **Si** ($Cat_g(W_i)$ ∈ {NN, NNS}) et ($Geo_{ERC}(W_i)$ = valeur) **Alors**
11. $Wc_{conditions} \leftarrow (Geo_{ERC}(W_i) = \text{`}W_i\text{'})$

//Heuristique 4

8

12. **Si** ($Cat_g(W_i) \in$ {NN, NNS, NNP, NNPS}) et ($Cat_g(W_{i+1}) \in$ {JJR}) **Alors** //JJR: adjectif comparatif
13. *Calculer la similarité entre les mots : (W_{i+1}), Inferior et Superior*
14. **Si** Similarité (W_{i+1},"Inferior") ≥ seuil **Alors**
15. $Wc_{conditions} \leftarrow W_i < 'W_{i+3}'$
16. **Sinon Si** Similarité (W_{i+1}, "Superior") ≥ seuil **Alors**
17. $Wc_{conditions} \leftarrow W_i > 'W_{i+3}'$
18. **Fin Si**
19. **Fin Pour**
20. **Fin**

Au niveau de cet algorithme, les trois premières heuristiques consistent à vérifier si un mot i correspond à une catégorie grammaticale bien déterminée, alors la clause WHERE représente une égalité soit entre la classe d'une entité (mot) i et le mot i ($Geo_{ERC}(W_i)$ = 'W_i'), soit entre deux mots (W_i = 'W_{i+2}'). En ce qui concerne l'heuristique 4, nous supposons que le mot W_i correspond à une des catégories grammaticales suivantes {NN, NNS, NNP, NNPS} et qu'il est suivi par un adjectif comparatif (W_{i+1}). Dans ce cas, pour déterminer la condition relative à la clause WHERE, nous proposons de calculer la similarité sémantique entre l'adjectif (W_{i+1}) et les mots *Inferior* et *Superior.* Si la similarité entre W_{i+1} et le mot « *inferior* » est supérieur à un seuil donné alors le nom (NN ou NNS) ou le nom propre (NNP ou NNPS) W_i est inférieur à la valeur W_{i+3} puisque W_{i+2} doit être obligatoirement une préposition qui relie l'adjectif W_{i+1} et la valeur W_{i+3} ($Wc_{conditions} \leftarrow W_i < 'W_{i+3}'$). Sinon si la similarité entre W_{i+1} et le mot « *superior* » est supérieur à un seuil donné alors W_i est supérieur à la valeur W_{i+3} dans ce cas ($Wc_{conditions} \leftarrow W_i > 'W_{i+3}'$).

3.2.3. Construction de la clause HAVING

Différents agrégats peuvent être appliqués pour utiliser les opérateurs d'agrégation spatiaux (SA_{op}) : area, perimeter, distance, etc. Opérateurs d'agrégation temporels (TA_{op}) d'Allen (Allen, 1983) : duration, intersect, equal, etc. Enfin des opérateurs d'agrégation spatio-temporels (STA_{op}) : moving-distance, etc. Le processus de construction de la clause HAVING (Hc) repose principalement sur deux heuristiques. Ces heuristiques sont dédiées à l'identification des attributs d'agrégation encapsulés dans des adjectifs comparatifs ou superlatifs. La première heuristique dans l'algorithme 3 se concentre sur l'idée que si W_i est un nom ou un nom propre suivi d'un adjectif comparatif (noté JJR), les attributs d'agrégation (SA_{att}, TA_{att}, STA_{att}) doivent être identifiés. Dans ce cas, nous proposons d'utiliser la similarité sémantique de Wu-Palmer (Wu et Palmer, 1994) pour calculer la similarité entre W_i et les opérateurs d'agrégation.

Algorithme 3. Construction de la clause HAVING (Heuristique 1)

1. **Entrée** *QU = {w_1, w_2, w_3...,w_n },* SA_{op}, TA_{op}, STA_{op} // SA_{op} : Opérateur d'agrégation Spatiale, TA_{op} : Opérateur d'agrégation Temporel, STA_{op} : Opérateur d'agrégation Spatio-Temporel

2. **Sortie** $Hc_{conditions}$ \\ $Hc_{conditions}$:la liste des conditions de Hc
3. **Début**
4. **Pour tout** $W_{i \in \{1, 2, 3, n\}}$ **Faire**
5. **Si** ($Cat_g(W_i) \in$ {NN, NNS, NNP, NNPS}) suivie de ($Cat_g(W_j) \in$ {JJR}) **Alors**
6. **Pour tout** (SA_{op}) **Faire**
   7. **Si** la similarité (W_i, ($SA_{op(k)}$)) ≥ seuil **Alors**
      8. $SA_{att} = SA_{op(k)}(W_i)$ //SA_{att} : attribut d'agrégation spatial
   9. **Si** la similarité (W_j, "Inferior") ≥ seuil **Alors**
      10.$Hc_{conditions} \leftarrow SA_{op(k)}(W_i) < W_{j+2}$
   11. **Sinon Si** la similarité (W_j, "Superior") ≥ seuil **Alors**
      12.$Hc_{conditions} \leftarrow SA_{op(k)}(W_i) > W_{j+2}$
   13. **Fin Si**
14. **Fin Si**
15. **Fin Pour**
16. **Pour tout** (TA_{op}) **Faire**
   17. **Si** la similarité (W_i, ($TA_{op(l)}$)) ≥ seuil **Alors**
      18.$TA_{att} = TA_{op(l)}(W_i)$ //TA_{att}: attribut d'agrégation temporel
   19. **Si** la similarité (W_j, "Inferior") ≥ seuil **Alors**
      20.$Hc_{conditions} \leftarrow TA_{op(l)}(W_i) < W_{j+2}$
   21. **Sinon Si** la similarité (W_j, "Superior") ≥ seuil **Alors**
      22.$Hc_{conditions} \leftarrow TA_{op(l)}(W_i) > W_{i+2}$
   23. **Fin Si**
   24. **Fin Si**
25. **Fin Pour**
26. **Pour tout** (STA_{op}) **Faire**
   27. **Si** la similarité (W_i, ($STA_{op(m)}$)) ≥ seuil **Alors**
      28.$STA_{att} = STA_{op(m)}(W_i)$ //STA_{att}: attribute d'agrégation spatio-temporal
   29. **Si** la similarité (W_j, "Inferior") ≥ seuil **Alors**
      30.$Hc_{conditions} \leftarrow STA_{op(m)}(W_i) < W_{j+2}$
   31. **Sinon Si** la similarité (W_j, "Superior") ≥ seuil **Alors**
      32.$Hc_{conditions} \leftarrow STA_{op(m)}(W_i) > W_{i+2}$
   33. **Fin Si**
   34. **Fin Si**
35. **Fin Pour**
36. **Fin Si**
37. **Fin Pour**
38. **Fin**

La seconde heuristique (Algorithme 4) indique que si W_i est un adjectif superlatif (noté JJS) suivi d'un nom (NN/NNS), il est nécessaire d'utiliser les fonctions d'agrégation MAX() et MIN() dans une sous-requête.

Algorithme 4. Construction de la clause HAVING (Heuristique 2)

---

1. **Entrée** QU = $\{w_1, w_2, w_3 \ldots, w_n\}$, SA_{op}, TA_{op}, STA_{op}, $Wc_{conditions}$, Fc \\ Fc: la clause FROM
2. **Sortie** $Hc_{conditions}$
3. **Début**
4. **Pour tout** $W_{i \in \{1, 2, 3, n\}}$ **Faire**
5. **Si** ($Cat_g(W_i) \in$ {JJS}) suivie de ($Cat_g(W_j) \in$ {NN, NNS}) **Alors**
   6. **Si** la similarité (W_i, "Inferior") ≥ seuil **Alors**

10

7. $Hc_{conditions} \leftarrow W_j$ = Select MIN(W_j From Fc Where $Wc_{conditions}$)
8. **Sinon Si** la similarité (W_i, "Superior") ≥ seuil **Alors**
9. $Hc_{conditions} \leftarrow W_j$ = Select MAX(W_j From Fc Where $Wc_{conditions}$)
10. **Sinon**
11. **Pour tout** (SA_{op}) **Faire**
12. **Si** la similarité (W_j, ($SA_{op(k)}$)) ≥ seuil **Alors**
13. $SA_{att} = SA_{op(k)}(W_j)$
14. **Si** la similarité (W_i, "Inferior") ≥ seuil **Alors**
15. $Hc_{conditions} \leftarrow SA_{att}$ = Select MIN(SA_{att} From Fc Where $Wc_{conditions}$)
16. **Sinon si** la similarité (W_i, "Superior") ≥ seuil **Alors**
17. $Hc_{conditions} \leftarrow SA_{att}$ = Select MAX(SA_{att} From Fc Where $Wc_{conditions}$)
18. **Fin si**
19. **Fin si**
20. **Fin pour**
21. **Pour tout** (TA_{op}) **Faire**
22. **Si** la similarité (W_j, ($TA_{op(l)}$)) ≥ seuil **Alors**
23. $TA_{att} = TA_{op(l)}(W_j)$
24. **Si** la similarité (W_i, "Inferior") ≥ seuil **Alors**
25. $Hc_{conditions} \leftarrow TA_{att}$ = Select MIN(TA_{att} From Fc Where $Wc_{conditions}$)
26. **Sinon Si** la similarité (W_i, "Superior") ≥ seuil **Alors**
27. $Hc_{conditions} \leftarrow TA_{att}$ =Select MAX(TA_{att} From Fc Where $Wc_{conditions}$)
28. **Fin Si**
29. **Fin Si**
30. **Fin Pour**
31. **Pour tout** (STA_{op}) **Faire**
32. **Si** la similarité (W_j, ($STA_{op(m)}$)) ≥ seuil **Alors**
33. $STA_{att} = STA_{op(m)}(W_j)$
34. **Si** la similarité (W_i, "Inferior") ≥ seuil **Alors**
35. $Hc_{conditions} \leftarrow STA_{att}$=Select MIN($STA_{att}$ From Fc Where $Wc_{conditions}$)
36. **Sinon Si** la similarité (W_i, "Superior") ≥ seuil **Alors**
37. $Hc_{conditions} \leftarrow STA_{att}$=Select MAX($STA_{att}$ From Fc Where $Wc_{conditions}$)
38. **Fin Si**
39. **Fin Si**
40. **Fin Pour**
41. **Fin Si**
42. **Fin Pour**
43. **Fin**

3.2.4. Construction de la clause GROUP BY

En fait, pour déterminer la clause GROUP BY (notée GByc), les attributs doivent appartenir à la liste des attributs de l'instruction SELECT et différents des attributs d'agrégation (SA_{att}, TA_{att}, STA_{att}) (Algorithme 5).

Algorithme 5. Construction de la clause GROUP BY

1. **Entrée** $Sst_{att}L$ // La liste des attributs de l'instruction SELECT
2. **Sortie** $GByc_{att}L$ // liste des attributs de la clause GROUP BY
3. $GByc_{att}L \leftarrow \varnothing$
4. **Début**

5. **Pour tout** $W_{i \in \{1, 2, 3, n\}}$ **Faire**
   // W_i OU $Geo_{ERC}(W_i) \in Sst_{att}L$: le mot i (W_i) OU la classe du mot i ($Geo_{ERC}(W_i)$) appartient à la liste des attributs de l'instruction SELECT
   // W_i OU $Geo_{ERC}(W_i) \notin \{SA_{att}, TA_{att}, STA_{att}\}$: le mot i (W_i) OU la classe du mot i ($Geo_{ERC}(W_i)$) est différent des attributs d'agrégation (SA_{att}, TA_{att}, STA_{att})
6. **Si** (W_i OU $Geo_{ERC}(W_i) \in Sst_{att}L$) et ($W_i$ OU $Geo_{ERC}(W_i) \notin \{SA_{att}, TA_{att}, STA_{att}\}$) **Alors**
7. Ajouter le mot (W_i OU $Geo_{ERC}(W_i)$) à $GByc_{att}L$
8. **Fin Si**
9. **Fin Pour**
10. **Fin**

3.2.4. Génération de la requête finale

Les différentes clauses sont déterminées sans faire référence à une ou à plusieurs tables de base de données. Les tables sont obligatoires et font partie intégrante de la requête. L'objectif principal de cette étape est de construire la clause FROM (algorithme 6) et de réaliser la correspondance entre les attributs des clauses générées et les attributs d'une base de données géographique donnée. Pour ce faire, nous proposons d'utiliser une mesure de similarité sémantique pour assurer cette correspondance entre les attributs. Dans ce cas, nous adoptons la mesure de similarité de Wu et Palmer (1994) permettant de calculer la similarité sémantique et la relation sémantique entre les mots en considérant leur longueur de chemin dans la taxonomie WordNet (Miller, 1995). Cette mesure est basée sur Least Common Subsumer (LCS) qui représente l'ancêtre commun le plus spécifique de deux attributs :

$$Sim_{wp}(Att_{1,} Att_2) = \frac{2depth(LCS(Att_1, Att_2))}{depth(Att_1) + depth(Att_2)} \quad (1)$$

depth(LCS(Att_1; Att_2)) représente le nombre de relations IS-A de l'attribut le plus courant au racine. depth (Att_1) et depth (Att_2) représentent le nombre de relations IS-A à partir de chaque attribut Att_1 et Att_2 jusqu'au racine de la taxonomie (Guessoum et al., 2016).

Algorithme 6. Construction de la clause FROM

1. **Entrée** $Sst_{att}L$, $Wc_{att}L$, $Hc_{att}L$, BDG \\ $Hc_{att}L$: la liste des attributs de la clause HAVING, BDG : base de données géographique
2. **Sortie** $Fc_{att}L$ // liste des attributs de la clause FROM
3. $Fc_{att}L \leftarrow \varnothing$
4. **Début**
5. **Pour tout** Table ∈ BDG **Faire**
6. Calculer la similarité sémantique entre les attributs de la table i et les attributs de l'instruction SELECT et des clauses HAVING et WHERE ($Sst_{att}L$, $Wc_{att}L$, $Hc_{att}L$).
7. **Si** la similarité ≥ seuil **Alors**
8. Ajouter la $Table_i$ à $Fc_{att}L$
9. Remplacer les attributs des différentes clauses ($Sst_{att}L$, $Wc_{att}L$, $Hc_{att}L$) par les attributs de la $Table_i$

12

10. **Fin Si**
11. **Fin Pour**
12. **Fin**

Si plusieurs tables sont utilisées, la clause FROM (notée Fc) pourrait être une opération de jointure qui regroupe des tables appariées. En ce qui concerne la construction de GByc, nous proposons de remplacer le $GByc_{att}$ par ceux modifiés dans le Sst. Dans certains cas, les tables de base de données peuvent contenir des noms de colonnes ambigües. Il est donc difficile de trouver une similarité sémantique ou même une relation sémantique entre les attributs de requête générée et les attributs des tables de base de données. Afin de résoudre ce problème, nous proposons d'assurer la correspondance en mesurant la relation sémantique entre les attributs de (Sst, Wc, GByc, Hc) et les valeurs que peut prendre chaque attribut des tables de base de données. L'algorithme 7 présente les détails de ce processus.

Algorithme 7. Correspondance entre les attributs de la requête (ST-SQL) et les valeurs que chaque attribut des tables de la BDG peut prendre

1. **Entrée** $Sst_{att}L$, $Fc_{att}L$, $Wc_{att}L$, $GByc_{att}L$, $Hc_{att}L$, V_1Att_i // V_1Att_i: La première valeur associée à chaque attribut de la $Table_i$.
2. **Sortie** ST-SQL \\ SQL spatio-temporelle
3. **Début**
4. **Pour tout** Table ∈ BDG **Faire**
5. Calculer la relation sémantique entre les attributs de ($Sst_{att}L$, $Wc_{att}L$, $GByc_{att}L$, $Hc_{att}L$) et V_1Att_i
6. **Si** relation sémantique ≥ seuil **Alors**
7. Remplacer les attributs de ($Sst_{att}L$, $Wc_{att}L$, $GByc_{att}L$, $Hc_{att}L$) avec les attributs de la $Table_i$
8. Formuler la requête ST-SQL finale utilisant ($Sst_{att}L$, $Fc_{att}L$, $Wc_{att}L$, $GByc_{att}L$, $Hc_{att}L$)
9. **Fin Si**
10. **Fin Pour**
11. **Fin**

4. Résultats de simulation

Notre outil permet aux utilisateurs des SIG d'accéder facilement aux données en utilisant le langage naturel comme un moyen de communication. Dans ce travail, nous nous sommes intéressés par des données recueillies à partir de la base de données NATural DISasters (NATDIS)[1]. Ce dernier comprend des informations sur les catastrophes naturelles dans le monde entier depuis janvier 2001. Le processus de prétraitement est déclenché une fois qu'un utilisateur introduit sa question. Ce processus consiste à exploiter les techniques de TALN. À cette fin, nous avons utilisé le Stanford CoreNLP (Manning et al., 2014) pour assurer l'étiquetage morphosyntaxique, extraire les relations de dépendance et reconnaître les entités et les relations géographiques. Plus précisément, pour assurer la reconnaissance des

1. http://www.catnat.net

entités et des relations géographiques, nous avons adopté le Framework TokensRegex (Chang et Manning, 2014) inclus dans Stanford CoreNLP (Manning et al., 2014). Ainsi, ce framework fournit des annotations basées sur des expressions régulières. TokensRegex a été utilisé pour développer SUTime (Chang et Manning, 2012). Ce dernier est en fait basé sur un ensemble de règles utilisées pour assurer la détection des expressions temporelles. Une fois la question est prétraitée, le système est capable de traduire cette question en une requête spatio-temporelle en se basant sur un ensemble d'heuristiques (figure 2).

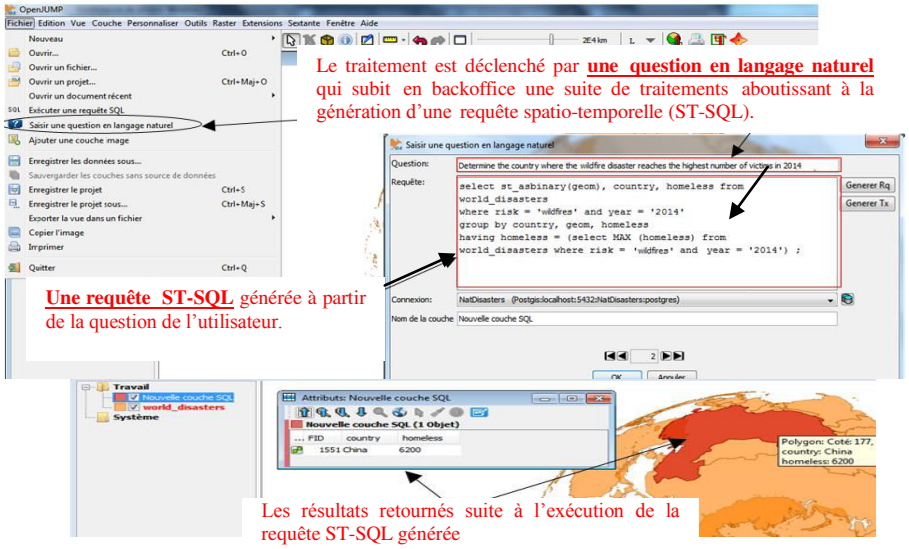


Figure 2. Génération d'une requête spatio-temporelle (ST-SQL)

5. Evaluation

Afin d'évaluer la performance de notre système à traduire une question en langage naturel sous une forme structurée à savoir une requête spatio-temprelle (ST-SQL) nous proposons d'utiliser les deux mesures bien connues : le Rappel et la Précision. En fait, nous proposons d'évaluer si les requêtes spatio-temporelles représentent bien la question. De ce fait, nous avons fourni un nombre de questions à des experts qui maitrisent bien le langage de requête SQL et plus précisément les requêtes géographiques pour qu'ils génèrent les requêtes adéquates. Dans ce cas, les résultats des requêtes générées par notre système bien évidement pour les mêmes questions seront comparés aux résultats retournés par les requêtes générées par les experts. En fait, quand un utilisateur introduit sa question, le système peut fournir différents types de résultats : soit des requêtes spatio-temporelles qui fournissent des réponses correctes en les comparant aux réponses des requêtes générées par les experts, soit des requêtes qui fournissent des réponses incorrectes à cause d'une mal traduction de la question en ST-SQL, soit, le système ne peut pas assurer la traduction de la question en requête spatiotemporelle. Notre système ne peut prendre

en considération que deux types de requêtes qui sont les requêtes spatio-temporelles simples et les requêtes spatio-temporelles d'intervalles. Les requêtes les plus sophistiquées ne sont pas encore prises en compte. Dans ce contexte, les deux mesures de rappel et de précision peuvent êtres définis comme suit : le rappel peut être défini comme étant le nombre de requêtes fournissant des réponses correctes, rapporté au nombre de requêtes spatio-temporelles générées. La précision représente le nombre de requêtes générées fournissant des réponses correctes, rapporté au nombre total de questions proposées.

$$\text{Rappel} = \frac{\text{Nombre de requêtes fournissant des réponses correctes}}{\text{Nombre de requêtes (ST-SQL) générées}} \quad (2)$$

$$\text{Précision} = \frac{\text{Nombre de requêtes fournissant des réponses correctes}}{\text{Nombre total de questions proposées}} \quad (3)$$

Dans le même contexte, nous avons adopté une mesure supplémentaire utilisée pour évaluer l'efficacité du système appelée Willingness (Minock, 2010). Cette mesure peut être définie comme étant le nombre de requêtes (ST-SQL) fournissant des réponses correctes et incorrectes, rapporté au nombre total de questions.

$$\text{Willingness} = \frac{\text{Nombre de requêtes fournissant des réponses correctes et incorrectes}}{\text{Nombre total de questions proposées}} \quad (4)$$

En se basant sur ces mesures, un ensemble de questions formulées en langage naturel (300 questions) ont été proposé. Les requêtes générées sont vérifiées manuellement, alors nous avons constaté différents types de réponses sont retournées lors de l'exécution de ces requêtes. Les résultats sont donnés dans le tableau suivant :

Tableau 1. Résultats expérimentaux

Le nombre total de questions	300
Le nombre de requêtes spatio-temporelles générées par notre système	269
Nombre de questions que le système est incapable de les traduire en requêtes spatio-temporelles	31
Nombre de requêtes fournissant des réponses correctes	232
Nombre de requêtes fournissant des réponses incorrectes	37
Précision	86.2%
Rappel	77.3%
Willingness	89%

D'après les taux de rappel (77.3%) et de précision (86.2%) obtenus, et en comparaison avec les requêtes spatio-temporelles générées par des experts nous pouvant constater que notre système a réussi à atteindre des résultats satisfaisants. Ainsi, le taux de Willingness obtenu est environ 89%, ce qui confirme la performance et l'efficacité de notre système.

5. Conclusion

Dans ce présent article, nous avons proposé une approche de génération de requêtes spatio-temporelles à partir des questions en langage naturel. D'abord une analyse profonde de ces questions est assurée afin de les traduire en un format structuré. Pour ce faire, nous avons eu recours à des techniques de TALN permettant le prétraitement de la question. Ensuite, nous avons proposé un ensemble d'heuristiques sur lesquelles est basé notre processus de génération des requêtes ST-SQL. Ainsi, une mesure de similarité est utilisée afin d'assurer la correspondance entre la question et la requête. Les expérimentations ont approuvé la validité de notre approche. Notamment, nous avons prouvé, que le fait d'exprimer des besoins d'information utilisant le langage naturel plutôt qu'un langage formel, pouvait évidemment fournir des résultats d'une qualité satisfaisante en assurant un confort supérieur pour le grand public. En guise de perspectives, notre approche peut être améliorée en ajoutant d'autres clauses permettant d'améliorer la recherche de l'information. Ainsi, nous envisageons d'étudier d'autres classes de questions spatio-temporelles, à savoir : les requêtes de comportement spatio-temporel et les requêtes de relation spatio-temporelles.

Références

Abacha A. B., (2012). *Recherche de réponses précises à des questions médicales : le système de questions-réponses MEANS.* Thèse en Informatique. Université Paris Sud - Paris XI.

Alexander R., Rukshan P., Mahesan S., (2013). Natural Language Web Interface for Database (NLWIDB). *In CoRR*

Allen J. F. (1983). Maintaining knowledge about temporal intervals. *Communications of the ACM*, p. 832–843.

Androutsopoulos I., Ritchie G., Thanisch P., (1995). Natural Language Interfaces to Databases - An Introduction. *In Journal of Natural Language Engineering*, vol 1, p 29–81.

Chang A. X., Manning C. D. (2012). Sutime: A library for recognizing and normalizing time expressions. *In: 8th International Conference on Language Resources and Evaluation*.

Chang, A. X., Manning C. D., (2014). *Tokensregex: Defining cascaded regular expressions over tokens,* Stanford University Technical Report.

Chaudhari P. P. (2013). Natural Language Statement to SQL Query Translator. *In International Journal of Computer Applications,* p. 18–22.

Chen W. (2014). Parameterized Spatial SQL Translation for Geographic Question Answering. *In Semantic Computing (ICSC), 2014 IEEE International*, p. 23–27

Deshpande A. K., Devale P. R., (2012). Natural Language Query Processing Using Probabilistic Context Free Grammar. *In International Journal of Advances in Engineering and Technology*, vol 3, p. 568–573.

Ferraro, J. P., Daumé H., DuVall S. L., Haug P. J., Harkema H., Chapman W., (2013). Improving performance of natural language processing part-of-speech tagging on clinical

narratives through domain adaptation. *Journal of the American Medical Information Association.*

Gesbert N. (2004). Formalisation des spécifications de bases de données géographiques pour une meilleure compréhension des données. *XXIIème congrès INFORSID*, p. 25–28.

Giordani A., Moschitti A., (2009). Semantic Mapping Between Natural Language Questions and SQL Queries via Syntactic Pairing. *In Natural Language Processing and Information Systems*, p. 207–221.

Giordani A., Moschitti A., (2012). Translating Questions to SQL Queries with Generative Parsers Discriminatively Reranked. *In COLING 2012: Posters*, p. 401–410.

Guessoum D., Miraoui M, Tadj C., (2016). A modification of wu and palmer semantic similarity measure. *In the Tenth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies.*

Manning, C. D., Surdeanu M, Bauer J., (2014). The stanford corenlp natural language processing toolkit. *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Marneffe M. C., MacCartney B., Manning C. D., (2006). Generating typed dependency parses from phrase structure parses. *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC).*

Mikolov T., Sutskever I., Chen K., Corrado G., Dean J. (2013). Distributed representations of words and phrases and their compositionality. *In Proceedings of the 26th International Conference on Neural Information Processing Systems, NIPS13*, p. 3111–3119, USA.

Miller G. A. (1995). Wordnet: A lexical database for english. *Communications of the ACM*.

Minock M. (2010). C-Phrase: A system for building robust natural language interfaces to databases. *Journal Data and Knowledge Engineering*, p. 290-302.

Popescu A. M., Etzioni O., Kauta H., (2003). Towards a Theory of Natural Language Interfaces to Databases. *In Proceedings of the 8th International Conference on Intelligent User Interfaces*, p. 149–157.

Rao G., Agarwal C., Chaudhry S., Kulkarni N., Patil D. S., (2010). Natural Language Query Processing using Semantic Grammar. *In International Journal on Computer Science and Engineering,* p. 219–223.

Raza A. (2008). *Deriving spatiotemporal relations from simple data structure.* The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences.

Worboys M. F. (1992). A generic model for planar geographical objects. *International Journal of Geographical Information Systems.*

Wu Z., Palmer M., (1994). Verb semantics and lexical selection. *In 32nd Annual Meeting of the Association for Computational Linguistics.*

Yaghmazadeh N., Wang Y., Dillig I., Dillig T. (2017). Type and content-driven synthesis of sql queries from natural language.

Yuan M., McIntosh J., (2002). A typology of spatiotemporal information queries. *The Springer International Series in Engineering and Computer Science,* Springer, Boston.

TemporalEMF: A Temporal Metamodeling Framework - Extended Abstract

Abel Gómez [1], **Jordi Cabot** [1,2], **Manuel Wimmer** [3]

1. *Internet Interdisciplinary Institute (IN3), Universitat Oberta de Catalunya (UOC), Spain*
   agomezlla@uoc.edu
2. *ICREA, Spain*
   jordi.cabot@icrea.cat
3. *CDL-MINT, TU Wien, Austria*
   wimmer@big.tuwien.ac.at

KEYWORDS: *Temporal Models, Model-Driven Engineering*

Modeling tools and frameworks have improved drastically in the last decade due to the maturation of metamodeling concepts and techniques (Brambilla *et al*., 2017). A concern which did not yet receive enough attention is the temporal aspect of metamodels and corresponding models when it comes to model valid time and transaction time dimensions instead of just arbitrary user-defined times (Gregersen, Jensen, 1999).

Indeed, existing modeling tools provide direct access to the most current version of a model, but very limited support to inspect the model state at specific past time periods (Bill *et al*., 2018; Benelallam *et al*., 2017). This typically requires looking for a model version stored in some kind of model repository roughly corresponding to that time period and using it to manually retrieve the required data. This approximate answer is not enough in scenarios that require a more precise and immediate response to temporal queries like complex collaborative co-engineering processes or runtime models (Mazak, Wimmer, 2016).

To deal with these new scenarios, temporal language support must be introduced as well as an infrastructure to efficiently manage the representation of both historical and current model information. Furthermore, query means are required to validate the evolution of a model, to find interesting modeling states, as well as execution states. Using existing technology to tackle these requirements is not satisfactory as we later discuss.

To tackle these limitations, we reuse well-known concepts from temporal languages to propose a temporal metamodeling framework, called *TemporalEMF*, that adds native temporal support. In *TemporalEMF*, models are automatically treated as temporal, and temporal query support allows to retrieve model elements at any point in time. Our framework is realized on top of EMF (Steinberg *et al*., 2009).

Models history is transparently stored in a NoSQL database, thus supporting large evolving models. We evaluated the resulting framework using an Industry 4.0 case study of a production system simulator (Mazak *et al*., 2017). The results showed good scalability for storing and accessing temporal models without requiring changes to the syntax and semantics of the simulator.

Specifically, the contribution of this work is three-fold: (*i*) we present a **lightweight extension** of current metamodeling standards to build a temporal metamodeling language. This metamodel is also represented as a profile for augmenting existing metamodels with information about temporal aspects; (*ii*) we introduce an **infrastructure to manage temporal models** by combining EMF and HBase (The Apache Software Foundation, 2018), an implementation of Google's BigTable storage (Chang *et al*., 2006); and (*iii*) we outline a **temporal query language** to retrieve historical information from models as an extension of the well-known Object Constraint Language.

Please note that contributions do not change the general way how models are used: if only the latest state is of interest, the model is transparently accessed and manipulated in the standard way as offered by the EMF. Thus, all existing tools are still applicable, and the temporal extension is considered to be an add-on.

As further work, we plan to extend *TemporalEMF* in several directions. At the modeling level, we will predefine some useful temporal patterns to facilitate the definition of temporal queries and operations. At the technology level, we will explore the integration of our infrastructure in other NoSQL backends and Web-based modeling environments to expand our potential user base. Finally, we aim to exploit the generated temporal information for a number of learning and predictive tasks to improve the user experience with modeling tools. For instance, we could classify users based on their typical modeling profile and dynamically adapt the tool based on that behaviour.

References

Benelallam A., Hartmann T., Mouline L., Fouquet F., Bourcier J., Barais O. *et al*. (2017). Raising time awareness in model-driven engineering: Vision paper. In *Proc. of models*, p. 181-188.

Bill R., Mazak A., Wimmer M., Vogel-Heuser B. (2018). On the need for temporal model repositories. In *Proc. of staf workshops*, pp. 136–145.

Brambilla M., Cabot J., Wimmer M. (2017). *Model-driven software engineering in practice, 2nd edition*. Morgan & Claypool Publishers.

Chang F., Dean J., Ghemawat S., Hsieh W. C., Wallach D. A., Burrows M. *et al*. (2006). Bigtable: A distributed storage system for structured data. In *Proc. of osdi*, pp. 15–15.

Gregersen H., Jensen C. S. (1999). Temporal entity-relationship models - A survey. *IEEE Trans. Knowl. Data Eng.*, Vol. 11, No. 3, pp. 464–497.

Mazak A., Wimmer M. (2016). Towards liquid models: An evolutionary modeling approach. In *Proc. of cbi*, pp. 104–112.

Mazak A., Wimmer M., Patsuk-Boesch P. (2017). Reverse engineering of production processes based on Markov chains. In *Proc. of case*, p. 680-686.

Steinberg D., Budinsky F., Paternostro M., Merks E. (2009). *EMF: Eclipse Modeling Framework 2.0* (2nd ed.). Addison-Wesley Professional. (ISBN: 0321331885)

The Apache Software Foundation. (2018). *Apache HBase.* (http://hbase.apache.org/)

Réduction de la quantité de données à visualiser dans l'OLAP multifonctions

Ali HASSAN[1], Patrice DARMON[1]

1. équipe R&D, Umanis
7-9 rue Paul Vaillant Couturier, 92300, Levallois-Perret, France
ahassan@umanis.com, pdarmon@umanis.com

Résumé. Dans le contexte d'OLAP multifonctions, le modèle conceptuel de l'entrepôt de données permet d'associer à une même mesure, une fonction d'agrégation différente pour chaque axe d'analyse, chaque hiérarchie et chaque niveau de granularité. Un système OLAP permet d'analyser les données multidimensionnelles interactivement à l'aide de tableaux croisés dynamiques et diagrammes. Malheureusement, aucun travail n'étudie la question de la lisibilité des diagrammes dans le contexte d'OLAP multifonctions. Donc, dans cet article, nous proposons une méthode de post-traitement des résultats des requêtes OLAP multifonctions afin d'améliorer la lisibilité en réduisant la quantité de données à visualiser. Cette méthode consiste à agréger les données vers des granularités moins détaillées. Autrement dit, faire un forage vers le haut (Rollup). La méthode proposée exploite d'abord le modèle conceptuel de l'entrepôt de données afin de trouver les agrégations qui ont été déjà réalisées dans la requête OLAP. Ensuite, à partir du résultat de la requête OLAP, en contrôlant la validité des fonctions d'agrégation, cette méthode trouve les opérations Rollup possibles et élimine celles qui sont interdites. Finalement, la méthode analyse les opérations Rollup possibles pour en choisir une qui donne à la fois des diagrammes lisibles et garde le maximum de détails possibles.

ABSTRACT. In the context of multifunction OLAP, the data warehouse conceptual model allows to associate to the same measure, a different aggregation function for each analysis axis, each hierarchy and each granularity level. An OLAP system allows to analyze interactively multidimensional data using pivot tables and diagrams. Unfortunately, no works investigate readability issues of diagrams in the context of multifunction OLAP. Therefore, in this article, we propose a method of post-processing of multifunction OLAP queries results to improve readability by reducing the amount of data to be visualized. This method aggregates the data to less detailed granularities. In other words, doing a Rollup. The proposed method first exploits the conceptual data warehouse model to find aggregations that have already been done in the OLAP query. Then, from the OLAP query result, by checking the aggregation functions validity, this method finds possible Rollup operations and eliminates those that are forbidden. Finally, the method analyzes possible Rollup operations to select the one that both gives readable diagrams and keeps as many details as possible.

Mots-clés : visualisation de données, OLAP, agrégation multifonctions, réduction de données.

KEYWORDS: data visualization, OLAP, multifunction aggregation, data reduction.

1. Introduction

L'analyse OLAP consiste à observer des indicateurs (mesures) selon plusieurs axes d'analyse (dimensions). Les décideurs utilisent des opérateurs OLAP (*Rollup* et *Drill-down*) afin d'analyser les mesures selon plusieurs niveaux de détails. Les données sont donc regroupées selon les niveaux sélectionnés et agrégés en utilisant des fonctions d'agrégation. Classiquement, l'OLAP offre la possibilité d'utiliser une seule fonction pour agréger une mesure dans tout l'espace multidimensionnel. Afin de surmonter cette limitation, *Hassan et al.* (2014) ont proposé un modèle assez expressif pour associer plusieurs fonctions d'agrégation à la même mesure. Par exemple, un tel modèle permet d'analyser la précipitation annuelle qui est calculée par la **somme** des précipitations quotidiennes. Cependant, la précipitation moyenne d'un département est calculée par la **moyenne** des précipitations de toutes les villes. Comme les fonctions généralement ne sont pas commutatives, le modèle proposé contrôle la validité du calcul en planifiant un ordre d'exécution entre les fonctions d'agrégation. Par exemple, il faut agréger les précipitations par la somme (avec un ordre d'exécution 1) sur la dimension 'Date' avant de les agréger par la moyenne (avec un ordre d'exécution 2) sur la dimension 'Géographie'.

Le résultat d'une requête OLAP est visualisé classiquement en utilisant un tableau croisé dynamique et des diagrammes (en barres, en secteurs, etc.). Par exemple, le tableau croisé dynamique en haut de la figure 1 montre la précipitation moyenne quotidienne (les deux premiers jours de janvier et février) par département. On peut apercevoir qu'une telle table a une lisibilité acceptable bien qu'elle affiche 120 données. L'affichage de même nombre de données en diagramme en barres peut réduire considérablement la lisibilité (figure 1 bas à gauche) ou voir devenir complètement illisible en utilisant un diagramme en secteurs (figure 1 bas à droit). Notre objectif est de proposer un système qui permet d'adapter automatiquement la quantité de données à afficher afin de garder la lisibilité des diagrammes utilisés.

Précipitation moyenne		Année	2010				2011				2012				2013				2014			
		Mois	janv		févr		janv		févr		janv		févr		janv		févr		janv		févr	
		jour	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2
Pays	Région	Département																				
France	Midi-Pyrénées	Haute-Garonne	12,38	13,75	13,75	12,88	11,63	11,38	11,63	0	14,38	13,5	12,38	11,38	11,88	13,38	11,75	11,38	13,63	14,13	11,63	11,25
		Hautes-Pyrénées	11,88	10,63	10,88	12,38	14,38	12,25	15,38	12	8,88	10	13,25	9	12,13	10,5	10,75	16,38	11,75	12,13	11,63	14
	Île-de-france	Paris	9,5	13	13,13	13,25	12,38	8,75	14,25	10,38	13,13	11,5	13,5	11,63	10,25	11,75	10,63	12,88	14,25	12,25	13,75	13,75
		Seine-et-Marne	12,75	11,25	12	11,38	13,75	11,88	13,5	13,5	11,5	14,38	10,75	12	12,38	12	13,5	13,88	15,13	13,63	13,25	12,88
	Rhône-Alpes	Rhône	11,63	11,38	12,5	11,88	11,25	12,75	16,13	14	11,38	11,63	11,63	13,38	10,75	12,38	11,63	13,25	11,38	13,13	11,25	12,5
		Isère	12,88	10,75	9,63	10,13	13,13	12,88	8,5	9,63	11,75	13,13	13,13	12,25	12	11,38	10,63	12,63	12,13	9,88	14,5	10,13

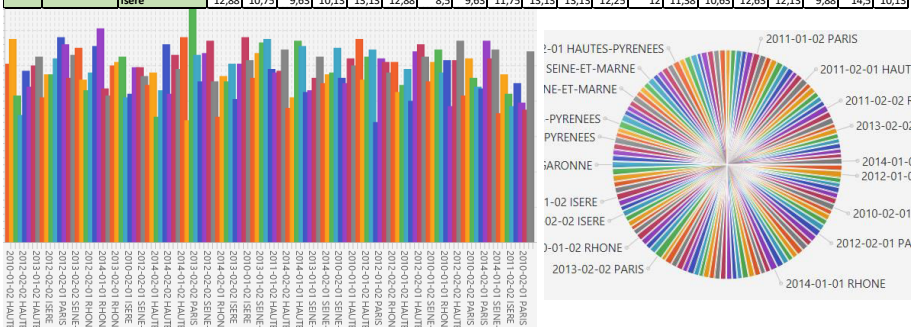


Figure 1. Visualisation des précipitations moyennes par jour et département

2. État de l'art

La plupart des travaux existant concernant la visualisation de grandes quantités de données (Wang *et al.*, 2017 ; Pahins *et al.*, 2017 ; Miranda *et al.*, 2018) ont pour objectif d'améliorer la performance du système en pré-agrégeant les données et en stockant les résultats en mémoire. La lisibilité des données visualisées a été traitée dans certains travaux. Les travaux : (Silva, *et al.*, 2012) pour les données spatiales, (Jugel *et al.*, 2014) pour les données temporelles et (Golfarelli *et al.*, 2014) pour la lisibilité du tableau croisé dynamique, proposent de réduire la quantité de données visualisées en agrégeant et regroupant (clustering) les données. Malheureusement, aucun de ces travaux n'étudie comment réduire la quantité de données dans le contexte d'OLAP multifonctions. Donc, dans cet article, nous proposons une méthode pour améliorer la lisibilité des résultats des requêtes OLAP multifonctions.

3. Méthode de réduction des données

Notre proposition est basée sur une méthode de post-traitement des résultats des requêtes OLAP multifonctions. Cette méthode réduit la quantité de données à visualiser en fonction du diagramme utilisé. Par exemple, les diagrammes en barres peuvent visualiser normalement plus de données que les diagrammes en secteurs. Ce qui veut dire que les diagrammes en secteurs nécessitent d'agréger (réduire) les données plus que les diagrammes en barres.

Cette méthode consiste à agréger les données automatiquement vers des granularités moins détaillées. Autrement dit, faire un forage vers le haut (Rollup). Cette méthode comprend plusieurs étapes :

1- Étudier la requête OLAP courante : notre méthode exploite le modèle de l'entrepôt de données afin de trouver l'ordre d'exécution maximal des fonctions d'agrégation qui ont déjà été réalisées au cours de la requête. La requête se représente par la mesure et les niveaux de granularité de chaque dimension sélectionnée pour observer la mesure. Par exemple, la requête de la figure 1 peut se représenter par la mesure 'précipitation moyenne' et les niveaux ['Jour', 'Département'].

La méthode cherche les fonctions d'agrégation utilisées pour agréger la mesure à chaque niveau de granularité de la requête courante. Une fois que ces fonctions sont trouvées, la méthode trouve l'ordre d'exécution maximal. Dans notre exemple de la figure 1, le niveau 'Jour' est le niveau le plus détaillé sur la dimension 'Date', donc il n'y a pas d'agrégation sur cette dimension. Sur la dimension 'Géographie', la précipitation est agrégée au niveau 'Département' à partir de 'Ville' par la moyenne qui a un ordre d'exécution 2 qui est donc l'ordre d'exécution maximal déjà réalisé.

2- Trouver les opérations Rollup possibles : afin de réduire la quantité de données à visualiser, nous proposons de réaliser un Rollup à partir du résultat de la requête sans interroger la base de données. En contrôlant la validité de calcul des fonctions d'agrégation, notre méthode trouve les opérations Rollup possibles et élimine celles qui sont interdites. Nous pouvons déterminer les opérations Rollup possibles par le produit cartésien de listes de niveaux de granularité possibles sur

chaque dimension. Un niveau de granularité est considéré comme "possible" si la condition d'ordre d'exécution est satisfaite. Selon cette condition, la fonction d'agrégation qui agrège la mesure au niveau concerné devrait avoir un ordre d'exécution égal ou supérieur à l'ordre d'exécution maximal déjà réalisé. Ainsi, notre méthode examine tous les niveaux supérieurs aux niveaux actuels de la requête.

Concernant la figure 1, l'ordre d'exécution maximal déjà réalisé est 2. Donc, les niveaux ('Région' et 'Pays') sur la dimension 'Géographie' sont possibles parce que la mesure est agrégée à ces niveaux par la moyenne qui a un ordre d'exécution 2 contrairement aux niveaux ('Mois' et 'Année') sur la dimension 'Date' qui sont interdits parce que la mesure est agrégée à ces niveaux par la somme qui a un ordre d'exécution 1. On ajoute les niveaux actuels de la requête ('Département' et 'Jour') aux listes de niveaux possibles correspondant avant de faire le produit cartésien. Ainsi, les listes de niveaux possibles sur les dimensions 'Date' et 'Géographie' sont respectivement ['Jour'] et ['Département', 'Région', 'Pays']. On enlève l'analyse actuelle ['Jour', 'Département'] du résultat du produit cartésien de ces listes. Donc, les opérations Rollup possibles trouvées sont ['Jour', 'Région'] et ['Jour', 'Pays'].

3- Calculer la taille des données pour toutes les opérations Rollup possibles : pour calculer la taille des données pour une opération Rollup, il suffit de multiplier le nombre de membres de ses niveaux de granularité. Nous pouvons trouver le nombre de membres de niveaux de granularité en analysant les en-têtes (en ligne et en colonne) du tableau croisé dynamique du résultat de la requête de base. Par exemple, en regardant la figure 1, nous constatons qu'il y a 20 jours (20 membres), 3 régions et 1 pays. Ainsi, la taille des données de deux opérations Rollup possibles résultant de l'étape précédente est calculée comme suit :

– La taille des données de ['Jour', 'Région'] = 20 × 3 = 60

– La taille des données de ['Jour', 'Pays'] = 20 × 1 = 20

4- Choisir une opération Rollup : notre méthode réduit la taille des données selon le type de diagramme utilisé (par exemple, diagrammes en barres ou en secteurs). Elle est contrôlée par des paramètres exprimant la taille maximale autorisée pour chaque type de diagramme. Comme la question de la lisibilité des données est relative (c'est-à-dire que ce qui est lisible pour une personne ne l'est pas forcément pour tout le monde), nous supposons que ces limites sont définies par l'utilisateur lui-même. Notre méthode choisit automatiquement une opération Rollup pour chaque type de diagramme qui à la fois donne un diagramme lisible (respecte la taille maximale autorisée) et garde le maximum de détails possible.

Par exemple, un utilisateur pourrait déterminer la taille maximale des diagrammes en barres et en secteurs par 70 et 25 respectivement. En d'autres termes, un diagramme en barres peut avoir au maximum 70 barres et un diagramme à secteurs peut avoir jusqu'à 25 secteurs. Dans ce cas, notre méthode sélectionne l'agrégation des données au niveau ['Jour', 'Région'] pour le diagramme en barres et au niveau ['Jour', 'Pays'] pour le diagramme en secteurs.

5- Réaliser l'opération Rollup choisie : une fois que les agrégations (les opérations Rollup) sont sélectionnées, elles doivent être exécutées. Ainsi, la méthode

devrait trouver les fonctions d'agrégation qui agrègent la mesure sur toutes les dimensions entre les niveaux actuels et les niveaux de l'opération Rollup choisie. Ensuite, les fonctions trouvées doivent être exécutées en fonction de leur ordre d'exécution. Si la fonction d'agrégation est algébrique, des valeurs intermédiaires doivent être stockées. Par exemple, pour éviter le calcul de la moyenne des moyennes, la fonction algébrique AVG nécessite de stocker les valeurs intermédiaires SUM et COUNT. Finalement, les diagrammes visualisent les données agrégées (cf. figure 2).

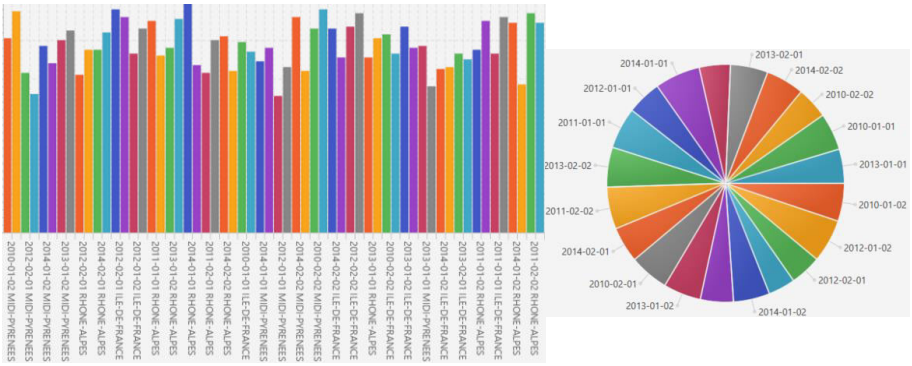


Figure 2. Visualisation des précipitations moyennes améliorée

4. Implémentation

Dans cette section, nous démontrons la faisabilité de notre proposition. Notre méthode est implémentée comme une extension du prototype 'OLAP-Multi-Functions' (*Hassan et al.,* 2015). Ce prototype a une architecture à deux niveaux (Figure 3). Le premier est les *interfaces*. Il est développé en Java. Il permet de créer et visualiser les schémas multidimensionnels multifonctions et d'interroger l'entrepôt de données. Le deuxième est le *stockage,* où un SGBDR (Oracle), est utilisé pour stocker le méta-schéma et l'entrepôt de données. Un générateur de requêtes SQL est développé en tant que procédures stockées. Il traduit, en prenant en compte le contexte multifonctions, les interactions des utilisateurs (requêtes OLAP) en requêtes SQL.

Notre méthode est développée en Java. Elle prend en entrée deux fichiers ('Données.JSON' et 'Mondrian.xml'). Le premier est le résultat de la requête OLAP en format JSON. Le deuxième est une extension du schéma Mondrian où nous décrivons l'association de plusieurs fonctions d'agrégation à la même mesure.

Les cinq étapes de la méthode sont exécutées séquentiellement (l'une après l'autre). Si plusieurs types de visualisation sont utilisés (par exemple, en barres, en secteurs), une exécution multiple des étapes 4 (choisir une opération Rollup) et 5 (réaliser l'opération Rollup choisie) est nécessaire. Ainsi, une exécution parallèle (un thread par type de visualisation) est utilisée. Les résultats de la réduction de données sont stockés par type de visualisation (par exemple, les fichiers 'Barre.JSON' et 'Secteur.JSON') avant d'être envoyés à la visualisation correspondante.

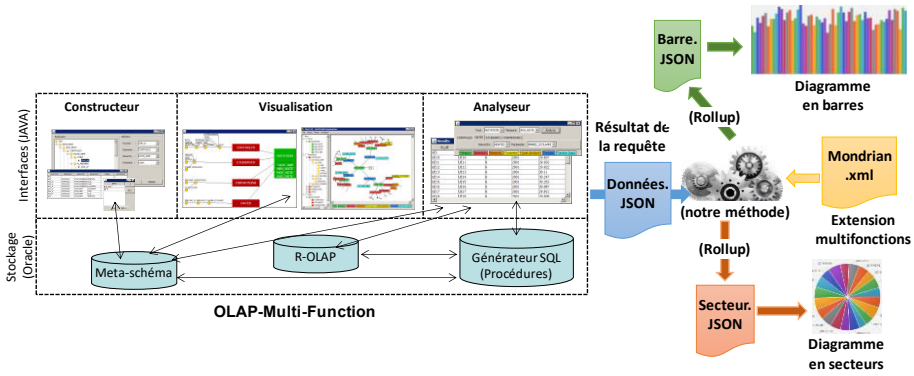


Figure 3. Implémentation de la méthode (Prototype)

5. Conclusion

Nous avons présenté une méthode de post-traitement pour améliorer la lisibilité des requêtes OLAP multifonctions. Elle permet la réduction automatique de la quantité de données selon le diagramme utilisé. Elle prend en compte la validité de calcul afin d'éviter les opérations d'agrégation de données (Rollup) interdites.

Bibliographie

Golfarelli M., Graziani S., Rizzi S. (2014). Shrink: An OLAP operation for balancing precision and size of pivot tables. *Data and Knowledge Engineering*, vol. 93, p. 19-41.

Hassan A., Ravat F., Teste O., Tournier R., Zurfluh G. (2014). Multidimensional database modelling with differentiated multiple aggregations. *JDS*, vol. 23, n° 4, p. 437-459.

Hassan A., Ravat F., Teste O., Tournier R., Zurfluh G. (2015). Differentiated Multiple Aggregations in Multidimensional Databases. *TLDKS*, vol. 21, p. 20-47.

Jugel U., Jerzak Z., Hackenbroich G. (2014). M4 : A visualization-oriented time series data aggregation. *PVLDB*, vol. 7, n° 10, p. 797–808.

Miranda F., Lins L., Klosowski J. T., Silva C. T. (2018). Topkube: A rank-Aware data cube for real-Time exploration of spatiotemporal data. *TVCG*, vol. 24, n° 3, p. 1394-1407.

Pahins C. A. L., Stephens S. A., Scheidegger C., Comba J. L. D. (2017). Hashedcubes: Simple, Low Memory, Real-Time Visual Exploration of Big Data. *TVCG*, vol. 23, n° 1, p. 671-680.

Silva R., Moura-Pires J., Santos M. Y. (2012). Spatial Clustering in SOLAP Systems to Enhance Map Visualization. *IJDWM*, , vol. 8, n° 2, p. 23-43.

Wang Z., Ferreira N., Wei Y., Bhaskar A. S., Scheidegger C. (2017). Gaussian Cubes: Real-Time Modeling for Visual Exploration of Large Multidimensional Datasets. *TVCG*, vol. 23, n° 1, p. 681-690.

Une méthode de conception collaborative d'entrepôt de données

Application à la biodiversité

Amir Sakka[1,2], Sandro Bimonte[1], Lucile Sautot[3], Guy Camilleri[2], Pascale Zaraté[2], Aurélien Besnard[4]

1. UR TSCF, IRSTEA
9 avenue Blaise Pascal, 63178 Aubière, France
prenom.nom@irstea.fr

2. IRIT, Université de Toulouse
prenom.nom@irit.fr

3. UMR TETIS, AgroParisTech
M. de la Télédétection, 500 rue Breton, 34000 Montpellier, France
lucile.sautot@agroparistech.fr

4. LPO Aquitaine
433 chemin de Leysotte, 33140 Villenave-d'Ornon, France
aurelien.besnard@lpo.fr

RÉSUMÉ. Dans le contexte de l'information géographique volontaire (VGI), les bénévoles ne participent pas aux processus décisionnels. De plus, les systèmes VGI n'offrent pas d'outils d'analyse avancés. C'est pourquoi, dans cet article, nous proposons d'utiliser les systèmes d'information décisionnels pour analyser les données VGI, et nous définissons une nouvelle méthode de conception de ces systèmes, qui permet d'impliquer les volontaires dans la définition des besoins analytiques. Nous le validons à l'aide d'une étude de cas sur la biodiversité.

Mots-clés : OLAP, Entrepôt de données, Information géographique volontaire, GDSS

Introduction

La VGI est "*la mobilisation d'outils pour créer, rassembler et diffuser des données géographiques fournies par des volontaires*" [18]. Elle permet de gérer des données géolocalisées (par exemple Openstreetmap[1]), et est largement utilisée dans différents domaines d'application : urbanisme, suivi de la biodiversité, caractérisation risques, etc. Habituellement, les volontaires sont des producteurs de données et des consommateurs passifs d'analyses de données, fournies par des organismes/entreprises. Ce paradigme "collecte des données *bottom-up* et analyse des données *top-down*" représente un obstacle important pour le développement des observatoires volontaires, car les producteurs de données se sentent exclus du processus décisionnel [13]. De plus, comme nous l'avons souligné dans [2], la VGI ne présente pas de fonctionnalités permettant d'analyser d'importants volumes de données géospatiales.

Les systèmes VGI sont conçus pour les tâches opérationnelles et l'analyse complexe de petits volumes de données spatiales, tandis que les systèmes SOLAP (Spatial On-Line Analytical Processing) sont plus pertinents pour l'analyse basée sur l'exploration d'ensembles massifs de données spatiales stockées dans un entrepôt de données [17, 12]. Puisque les entrepôts de données sont conçus en fonction des sources de données et des besoins des utilisateurs, plus le modèle de l'entrepôt de données reflète les besoins des intervenants, plus ces derniers utiliseront leurs données [12, 15]. La mise à disposition d'applications décisionnelles adaptées aux besoins analytiques de la communauté VGI représentera un progrès important, puisque : (1) de nouvelles possibilités d'analyses efficaces sur des données volumineuses provenant d'observatoires différents seront possibles et (2) les volontaires seront de plus en plus motivés à collecter des données.

1. Méthode proposée

Nous proposons un nouveau système OLAP (OLAP 2.0). L'idée principale est de permettre aux volontaires d'exprimer séparément leurs besoins en matière d'analyse OLAP, dans un premier temps. Ces exigences seront ensuite traduites en modèles multidimensionnels (c.-à-d. en modèle d'entrepôt de données). Ces modèles multidimensionnels sont transformés automatiquement en prototypes grâce à la méthode ProtOLAP, afin de permettre aux volontaires une meilleure appropriation des concepts d'informatique décisionnelle. Ensuite, ces modèles multidimensionnels sont soumis à un ensemble de volontaires particuliers appelés *committers*, qui sont pleinement impliqués dans le projet et qui ont de l'expérience vis-à-vis des données collectées par les volontaires. Certains auteurs mettent l'accent sur la nécessité de l'intendance des données (menée par les *committers* dans notre approche) pour résoudre les problèmes liés au manque d'expérience des utilisateurs dans la spécification des requêtes et aux problèmes de propriété/sensibilité des données

[1] https://www.openstreetmap.org

rencontrés par les organisations au cours du processus d'implémentation des entrepôts de données. Ainsi, les *committers* décident de mettre en œuvre ou non des besoins exprimés (c'est-à-dire des modèles multidimensionnels) par les volontaires, en fonction de leur expertise pour juger de la pertinence des besoins. Ensuite, des informaticiens spécialisés en SI décisionnels sont chargés de la mise en œuvre des modèles choisis par les committers. Enfin, les nouveaux modèles multidimensionnels sont implémentés et mis à la disposition de tous les utilisateurs qui peuvent visualiser, explorer et analyser les données.

2. Expérimentation

Dans le cadre du projet VGI4Bio[2], nous mobilisons deux bases de données VGI (Visionature et l'Observatoire Agricole de la Biodiversité - OAB) pour construire des applications SOLAP d'analyse des indicateurs de biodiversité des terres agricoles. Visionature et OAB comptent respectivement 7682 et 1500 volontaires qui produisent des données. Parmi les utilisateurs potentiels intéressés par l'analyse de ces données, nous avons identifié un grand nombre d'utilisateurs appartenant à des catégories diverses telles que : les volontaires intéressés pour améliorer la qualité de leur production de données, leurs pratiques quotidiennes, etc., et des organismes publics et privés (DREAL, Chambre d'Agriculture, etc.).

Pour la validation expérimentale de notre proposition, nous avons engagé quatre volontaires avec des compétences différentes, et nous avons identifié quatre committers. Pour la validation de la première étape à l'aide de ProtOLAP, nous avons compté le nombre de rencontres entre volontaires et informaticiens et leur durée. Le temps d'implémentation d'un prototype d'entrepôt de données avec ProtOLAP est négligeable, puisqu'il ne prend que quelques minutes. En moyenne, il y a trois réunions par bénévole et chacune dure une heure. Par conséquent, nous pouvons conclure que ce n'est que lorsque le nombre de volontaires est faible que l'utilisation de la méthodologie ProtOLAP est possible. Lorsque le nombre de bénévoles devient important, une nouvelle méthodologie doit être fournie pour permettre aux bénévoles de définir eux-mêmes leurs modèles OLAP sans l'intervention des informaticiens.

Conclusion

Dans cet article, nous proposons une nouvelle méthode de conception collaborative qui permet d'impliquer les volontaires dans la définition des besoins d'analyse par rapport aux données VGI. Notre méthode permet aux volontaires non qualifiés en informatique de participer au processus de conception.

Nos travaux futurs porteront sur l'automatisation de la collecte des besoins auprès des volontaires, sans quoi notre méthode ne sera pas utilisable sur de larges communautés.

[2] www.vgi4bio.fr