

Les Besoins Fonctionnels Candidats à l'Entreposage et l'Analyse en Ligne

Zouhir Djilani*, Selma Khouri*, Ladjel Bellatreche *, Abderrahmane Khiat **

*LIAS/ISAE-ENSMA, Université de Poitiers, Futuroscope 86960
(prenom.nom)@ensma.fr

** Université d'Oran 1, Algérie
abderrahmane_khiat@yahoo.com

Résumé. Depuis peu d'années, la technologie des entrepôts et analyse des données (\mathcal{EDA}) est devenue mature. Cette maturité a poussé un nombre important d'entreprises à l'adopter. En analysant cette situation, nous constatons que les entrepôts ont souvent été *associés aux données*, qui représentent une mine d'or pour les entreprises. Mais d'autres entités capitales pour les entreprises présentent le même intérêt d'exploitation et d'analyse. Nous pouvons citer les exemples des modèles, des services, des processus, etc. Parallèlement, des efforts considérables ont été déployés pour intégrer d'autres entités que les données dans des référentiels (repositories), mais sans la vision d' \mathcal{EDA} . Combiner les deux solutions proposées par les référentiels et les \mathcal{EDA} serait un atout considérable à la fois pour les entreprises et les chercheurs. Dans cet article nous proposons une démarche complète de construction d'un entrepôt de besoins fonctionnels, souvent considérés comme une pré-condition pour concevoir tout type d'entrepôt ou de référentiel.

1 Introduction

Les solutions de persistance des données, à leur tête les bases de données (\mathcal{BD}), sont devenues des éléments incontournables pour la communauté industrielle et académique. Avec l'explosion des sources de données à l'intérieur et à l'extérieur des entreprises, la persistance a également contribué à matérialiser les données provenant de ces sources *fortement hétérogènes*¹ dans des *systèmes d'intégration* (\mathcal{SI}). Pour augmenter leur compétitivité, les entreprises ont fait appel à des outils d'analyse associés aux données, ce qui a donné le lieu aux entrepôts de données (\mathcal{ED}) et analyse en ligne (\mathcal{EDA}). Contrairement aux \mathcal{SI} , les \mathcal{ED} offrent une perception multidimensionnelle aux données exploitable par divers outils d'analyse, de visualisation et de navigation.

Historiquement, les \mathcal{ED} ont concerné les données traditionnelles. Ensuite ont émergé de nouveaux \mathcal{ED} pour prendre en compte d'autres types de données telles que les données XML, spatiales, sémantiques, tweets, données NoSQL, etc . Des outils d'analyse traditionnels ont

1. Cette hétérogénéité est syntaxique et sémantique.

également suivi cette évolution (par ex. XML-OLAP pour les données semi-structurées ou S-OLAP pour les données spatiales). En analysant finement le paysage de la technologie des \mathcal{ED} , nous souhaitons faire réunir des groupes de réflexion autour de deux phénomènes majeurs : (i) l'alliance entrepôt-données signifiant que les *entrepôts ont souvent été liés aux données structurées et semi-structurées*. Cela est justifié par la considération des données comme une mine d'or pour les entreprises pour des analyses. Récemment, cette alliance a été cassée avec l'arrivée d'un autre type de données issues des réseaux sociaux (tweets) souvent peu structurées, où les \mathcal{ED} ont été revisités pour prendre en compte ce type de données (Hannachi et al., 2013). Cette revisite a fait recours aux techniques et outils de traitement de la langue naturelle (TAL) (Hannachi et al., 2013). (ii) Les données alimentant les \mathcal{ED} sont souvent des données d'exploitation, associées au schéma interne des sources des données. Si nous faisons le parallèle avec les \mathcal{SI} présentés sous forme de référentiels, nous constatons que ces derniers sont plus polymorphes, du fait qu'ils sont associés à une large variété d'entités couvrant à la fois les données traditionnelles, avancées, des modèles, en passant par les API (Sun et al., 2015), les services (Belaid et al., 2009), les processus (Rosa et al., 2011) et les brevets (et al., 2012). Ces entités sont également précieuses pour les entreprises, car elles contiennent un savoir-faire important.

Dans cet article, nous croisons ces deux solutions (\mathcal{ED} et \mathcal{SI}) et nous tentons de résoudre un exercice sur la construction d'un entrepôt des besoins fonctionnels, considérés comme une pré-condition pour tout système d'entreposage des données. Dans notre laboratoire, nous avons travaillé depuis quelques années sur l'intégration des besoins hétérogènes dans le cadre des entreprises étendues comme Airbus. Dans une telle entreprise qui regroupe plusieurs partenaires géographiquement distribués (France, Allemagne, Espagne, Angleterre), les acteurs dans chaque site expriment leurs besoins en utilisant des formalismes (formalismes orientés buts, use case d'UML) et des vocabulaires fortement variés. Des solutions d'intégration basées sur des ontologies ont été proposées pour réduire l'hétérogénéité et uniformiser le vocabulaire de l'ensemble des partenaires (Boukhari, 2014; Khouri et al., 2014). Une particularité intéressante liée aux ontologies est qu'elles ont été utilisées dans l'ensemble des phases de construction et d'exploitation d'un \mathcal{ED} . Un point essentiel que nous souhaitons mettre en évidence est que la majorité des travaux sur les \mathcal{ED} sémantiques supposent l'existence d'une ontologie partagée référencée (que nous appelons hypothèse de l'ontologie unique) par l'ensemble des sources de données candidates à l'entreposage (Boukhari, 2014). Cette ontologie a été utilisée pour définir des opérateurs ETL au niveau sémantique. Dans notre étude nous considérons différentes hypothèses pour la construction de l'ontologie intégrant les sources (Wache et al., 2001) : (a) une ontologie unique, (b) des ontologies multiples, et (c) une ontologie partagée par les ontologies des sources. Dans le contexte des besoins fonctionnels, les hypothèses (b) et (c) sont plausibles. Cet article positionne une piste de réflexion autour de l'entreposage des besoins. L'article est structuré en quatre sections : la section 2 présente et compare les travaux connexes traitant des solutions d'intégration et de matérialisation des besoins proposées. La section 3 décrit l'ensemble des phases de construction de l' \mathcal{EBF} . La dernière section conclut l'article et donne quelques perspectives.

2 Travaux connexes

Le tableau 1 résume la revue de littérature des travaux les plus pertinents selon les trois axes connus des \mathcal{ED} : intégration, matérialisation et analyse. Nous avons commencé notre étude par

les travaux traitant l'intégration des besoins, comme (Wieringa et al., 1997; López et al., 2002; Nguyen et al., 2015). Ces travaux s'intéressent à une seule dimension d'intégration (intégration de formalismes (*If* dans le tableau) ou intégration conceptuelle (*IC*)), sans traiter la problématique de matérialisation des besoins. La dimension d'intégration linguistique (*IL*) est peu considérée. Concernant la dimension formalisme, ces travaux considèrent différents niveaux de formalisation (*Nf*) : informel (*I*), semi-formel (*SF*) et/ou formel (*F*). Ces travaux proposent des méthodes classiques ou ontologiques (Assawamekin et al., 2010). Nous remarquons dans le tableau 1 qu'aucune étude ne gère l'hétérogénéité des besoins sur les trois dimensions d'intégration simultanément (formalisme, conceptuel et linguistique). Nous nous sommes ensuite intéressés aux travaux proposant la matérialisation des besoins. Nous avons identifié plusieurs critères de comparaison : Persistance des besoins (*Pb*), solution d'entreposage (*Sep*) et Gestion de l'historique des besoins (*Hb*). Le tableau 1 montre que très peu d'études gèrent la persistance des besoins (Khouri et al., 2014; Boukhari, 2014), aucune étude ne considère l'entreposage et la gestion de l'historique des besoins. Le troisième axe concerne l'analyse des besoins, où nous avons identifié les critères suivants : Modélisation multidimensionnelle (*Am*), Analyses OLAP (*Olap*) et raisonnement sur les besoins (*Rb*). Nous remarquons que même les travaux récents traitant de la persistance des besoins ne traitent pas les analyses multidimensionnelles et OLAP des besoins. Par contre, plusieurs travaux proposent du raisonnement pour améliorer leur qualité. Notre solution regroupe les trois axes cités et traite l'ensemble des critères identifiés.

Dimension/ Travaux	Intégration				Matérialisation			Analyse		
	IC	IL	If	Nf	Pb	Sep	Hb	Am	Olap	Rb
Wieringa and al.(Wieringa et al., 1997)	✓	✗	✗	<i>I, SF, F</i>	✗	✗	✗	✗	✗	✗
Lopez and al.(López et al., 2002)	✗	✗	✓	<i>SF</i>	✗	✗	✗	✗	✗	✗
Brottier and al.(Brottier et al., 2007)	✗	✗	✓	<i>I</i>	✗	✗	✗	✗	✗	✓
Giorgini and al.(Giorgini et al., 2003)	✗	✗	✗	<i>SF</i>	✗	✗	✗	✗	✗	✓
Cognil and al.(Goknil et al., 2011)	✓	✗	✗	<i>F</i>	✗	✗	✗	✗	✗	✓
N.Assawamekin and al.(Assawamekin et al., 2010)	✓	✗	✗	<i>I</i>	✓	✗	✗	✗	✗	✗
TH Nguyen and al.(Nguyen et al., 2015)	✗	✗	✓	<i>SF</i>	✗	✗	✗	✗	✗	✓
CL Liu and al.(Liu, 2016)	✓	✗	✓	<i>SF</i>	✗	✗	✗	✗	✗	✓
Boukhari and al.(Boukhari, 2014)	✓	✗	✓	<i>SF</i>	✓	✗	✗	✗	✗	✓
Khouri and al.(Khouri et al., 2014)	✓	✗	✓	<i>SF</i>	✓	✗	✗	✗	✗	✓
Notre approche	✓	✓	✓	<i>SF</i>	✓	✓	✓	✓	✓	✓

TAB. 1 – Études des travaux connexes : comparaison et analyse

3 Approche proposée

La figure 1 présente l'architecture générale de notre approche. L'approche se base sur les structures ontologiques citées en introduction (ontologie partagée et ontologies multiples). L'approche repose ainsi sur les entrées suivantes : des sources de besoins qui décrivent l'ensemble des besoins selon un formalisme donné, utilisant les concepts d'une ontologie locale conceptuelle et les termes d'une ontologie locale linguistique. Ces ontologies référencent une ontologie globale (conceptuelle et linguistique) qui unifie les termes et concepts utilisés. Cette ontologie est soit existante (structure d'une ontologie partagée), soit construite par matching (ontologies multiples). Le modèle multidimensionnel de l' \mathcal{EBF} sera défini sur la base d'un

Les Besoins Fonctionnels Candidats à l'Entreposage et l'Analyse en Ligne

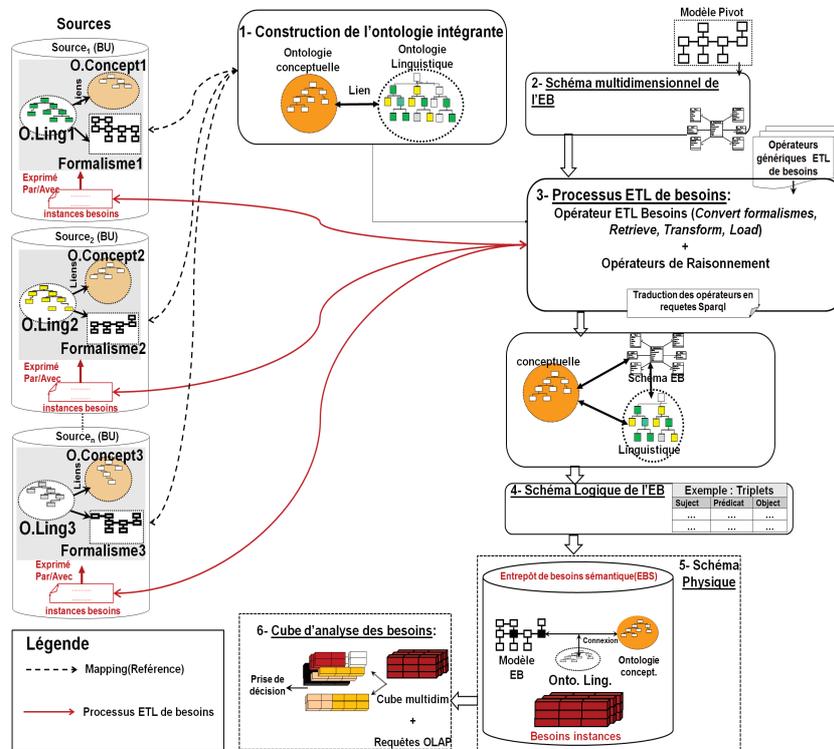


FIG. 1 – Architecture de notre démarche

modèle pivot unifiant les formalismes de modélisation des besoins. Un processus ETL adapté aux besoins permettra : d'extraire les besoins des sources, de les transformer (par ex. la conversion des formalismes sources au modèle de l' \mathcal{EBF} , le filtrage de certaines instances, l'union des instances, etc), ensuite de les charger dans le modèle de l' \mathcal{EBF} . Selon les mappings définis entre les ontologies locales et globale, le modèle de l' \mathcal{EBF} est ainsi connecté à l'ontologie globale, ce qui permettra ainsi de définir sémantiquement chaque composante du modèle. Nous présentons dans ce qui suit notre approche dans laquelle chaque étape consiste en un effort de réadaptation des fondements des entrepôts conventionnels aux entrepôts de besoins.

3.1 Construction de l'ontologie globale intégrante

La construction de l'ontologie intégrante repose sur les deux scénarios cités (ontologie multiple ou ontologie partagée). Les solutions d'entreposage ontologiques de manière générale suivent le deuxième scénario et supposent l'existence d'un schéma partagé entre les sources, ce qui limite leur autonomie. Nous proposons de relâcher cette contrainte par un processus de matching ontologique que requiert le premier scénario. Cette contribution a été proposée et validée dans (Djilani et al., 2016b). Nous résumons les principales étapes comme suit. Le processus de matching commence par le chargement des ontologies locales et calcule un en-

semble de mappings M entre les ontologies conceptuelles locales et M' entre les ontologies linguistiques. Comme résultat, le processus fusionne les ontologies conceptuelles et linguistiques locales en utilisant les mappings M et M' . Nous revenons ainsi au scénario de l'ontologie partagée. Le processus de matching commence par une étape de pré-traitement utilisant les techniques de traitement du langage naturel (conversion en minuscules, lemmatisation et élimination des mots d'arrêts) des termes utilisés pour la description des besoins. Le processus de matching effectue ensuite un matching aux deux niveaux : termes et concepts. Au niveau terminologique, des matchers linguistiques sont utilisés se basant sur le calcul de la similarité entre les termes des entités et aussi sur le calcul de similarité entre des chaînes de caractères (les noms des concepts, attributs, instances, commentaires, relations et labels). Trois mesures de similarité sont utilisées : la distance de *levenshtein*, la distance de *Jaro* et la distance de *SLIM-Winkler*. Pour réduire l'hétérogénéité conceptuelle, deux principales techniques sont utilisées : (i) une technique basée sur la structure interne des concepts utilisés et leurs propriétés (le domaine des entités, co-domaine, les cardinalités, le nombre d'attributs et les types de données qui composent les concepts). (ii) une technique utilisant la structure externe qui se base sur les relations entre les entités. Différentes expérimentations reposant sur le rappel et précision ont permis de valider ce processus de matching (Djilani et al., 2016b).

3.2 Modèle de l'entrepôt de besoins

Le modèle de l' \mathcal{EBF} doit refléter les entités caractérisant l'ensemble des besoins à entreposer. Le modèle de l' \mathcal{EBF} que nous proposons est illustré dans la figure 2. Ce modèle est conçu sur la base d'un premier modèle pivot proposé au laboratoire (Boukhari, 2014) et corrige ses limites. Le modèle que nous proposons ici permet de détailler toutes les composantes d'un besoin. Pour ce faire, nous nous sommes inspirés des vocabulaires contrôlés utilisés par les entreprises lors de la définition des besoins. Chaque besoin (*Requirement* dans la figure 2) précise ainsi une tâche détaillée en un triplet (sujet, verbe, objet). Un besoin est ainsi modélisé d'une façon plus proche de la représentation d'une phrase en langage naturel. D'autres entités sont également ajoutées (Criteria, Result, Actor, Relationship). Cette représentation détaillée des besoins facilite leur intégration et permet un raisonnement plus précis sur les besoins. Pour illustrer l'utilisation de ce modèle, nous l'instancions par l'exemple suivant : "*Le système permet de calculer le nombre des étudiants ayant une note en informatique supérieure à 10*". Selon le modèle, ce besoin est caractérisé par une action (*calculer*), un objet (*nb - Etudiant*), un sujet (*Système*). Ce besoin a donc comme tâche $\langle \text{Système}, \text{calculer}, \text{nombre Etudiant} \rangle$. Ce besoin a deux critères (*moyenne-etudiant > 10*) et (*module = informatique*). Ce besoin a comme résultat le "*nombre d'étudiants calculé*".

Le modèle multidimensionnel de l' \mathcal{EBF} est construit sur la base du modèle de besoins présenté (figure 3). Nous avons commencé par étendre ce modèle par deux nouvelles entités (*Time*, *Location*). L'entité *Time* représente le moment où le besoin a été exprimé au niveau du partenaire source. L'entité *Location* représente la localisation du partenaire source qui a exprimé ce besoin. Notre objectif est de permettre une analyse multidimensionnelle des besoins issus de diverses sources selon plusieurs dimensions. Pour ce faire, nous avons conçu le modèle multidimensionnel de façon à mettre l'entité "Requirement" au centre de cette analyse, représentant ainsi une entité Fait. Plusieurs mesures peuvent être définies pour ce fait comme le nombre (total, min, max) de besoins issus des sources. Les attributs de ce fait sont les attributs identifiés pour l'entité *Requirement* tels que : la description du besoins, sa priorité, etc.

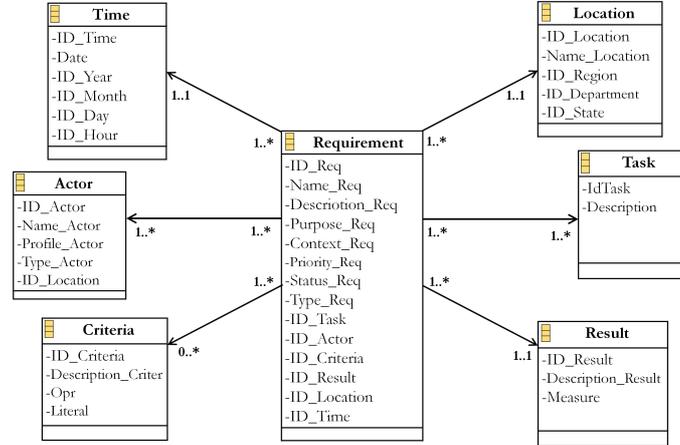


FIG. 3 – Le schéma en étoile de l'entrepôt de besoins

besoins. Pour définir ce processus ETL, nous nous sommes basés sur un ensemble d'opérateurs ETL définis pour les entrepôts conventionnels (Exp. Extract, Store, etc). Nous avons adapté ces opérateurs à notre contexte pour définir des opérateurs ETL orientés besoins. Ces opérateurs permettent l'extraction, la transformation et le chargement des besoins ainsi que des concepts et termes ontologiques utilisés. Par exemple, l'adaptation de deux opérateurs *Retrieve* et *Convert* se fait comme suit :

- **Retrieve** ($S_i, MF_i, B_j, C_j, TR_j$) : l'opérateur est chargé de la récupération du besoin B_j appartenant à une source S_i formalisée avec un formalisme MF_i , exprimé avec les classes C_j d'une ontologie conceptuelle locale et les termes TR_i d'une ontologie linguistique locale.

- **Convert** (S_i, B_i, fct, B_j) : cet opérateur permet la conversion du besoin $B_i \in S_i$ en B_j selon une fonction de conversion fct . Cet opérateur est utilisé pour les conversions classiques (comme la conversion de l'attribut date d'un format YYYY/MM/DD au format DD/MM/YYYY) et celles des formalismes. Par exemple, si le modèle source est un modèle de cas d'utilisation contenant une entité Action, une conversion de cette action permet de sauvegarder cette action dans l'entité Action du modèle de l' \mathcal{EBF} proposé. Nous avons définis et validé dans (Djilani et Khouri, 2015) un ensemble de règles de transformation d'un ensemble de modèles sources (utilisant des semi-formalismes) au modèle de l' \mathcal{EBF} .

Nous avons ensuite enrichi ces opérateurs ETL par des opérateurs de raisonnement sur les besoins. Trois opérateurs *Identification* ($B, S, IdentRule$), *Inference* ($B, S, InfRule$) et *CheckConsistency* ($B, S, ConsistenceCheckRule$) ont donc été définis pour gérer ce raisonnement. Ce raisonnement permet d'identifier des relations entre les besoins ainsi que les besoins inconsistants afin de les éliminer. Ces opérateurs se basent sur un ensemble de règles de raisonnement que nous avons défini dans (Djilani et Khouri, 2015). Puisque dans notre approche, les besoins sont connectés à l'ontologie, nous avons exploité le raisonnement ontologique afin de définir ces règles. Les principales étapes de l'algorithme ETL que nous avons définies sont les suivantes, pour chaque source de besoins :

- Extraire (opérateur *Retrieve*) les besoins B_j concernés par le mapping de la source S_i .

- Convertir (*Convert*) les besoins du format source vers le format intégrateur (*multidimensionnel*), sur la base des règles de transformation de modèles que nous avons définies.
- Filtrer (opérateur *Filter*) les besoins d'une source S_i , en appliquant une contrainte *CS*.
- Convertir les arguments des besoins si nécessaire en utilisant l'opérateur *Convert* et une fonction d'agrégation.
- Agréger les besoins en utilisant une fonction d'agrégation et l'opérateur *Aggregate*
- Stockage temporaire des besoins dans la staging area en utilisant l'opérateur *Store*
- Si deux besoins B_i et B_j qui doivent être unifiés sont :
 - pour la même source, l'algorithme utilise l'opérateur *merge* ;
 - pour les sources différentes, l'algorithme utilise l'opérateur *union*.
- Si deux besoins B_i et B_j présentent une relation entre eux, les unifier (opérateur *join*).
- Chargement (opérateur *Store*) des besoins sources B_j dans le schéma cible de l'entrepôt ;
- Raisonnement sur la base des opérateurs *Identification*, *Inference* et *CheckConsistency*, afin de détecter les relations complexes entre les besoins et de vérifier leur cohérence.
- Chargement des besoins transformés dans l'entrepôt de besoins

3.4 Connexion entre le modèle de l' \mathcal{EBF} et les ontologies

L'ensemble des éléments du modèle de l' \mathcal{EBF} sont définis sémantiquement en utilisant les deux couches de l'ontologie globale (conceptuelle et linguistique). Nous connectons d'abord le modèle de l' \mathcal{EBF} avec les éléments de l'ontologie conceptuelle. Les éléments du modèle de l' \mathcal{EBF} qui sont définis par des éléments ontologiques sont donc : les éléments de Task (Subject, Action et Object), Criteria et Result. La couche linguistique de l'ontologie globale est également présentée. La connexion se fait entre la couche conceptuelle et linguistique par un lien *hasLexicalEntry* entre les entités de l'ontologie conceptuelle et les termes de l'ontologie linguistique. De cette façon, les éléments des besoins utilisent des ressources ontologies de l'ontologie conceptuelle qui sont exprimés par des termes de l'ontologie linguistique.

3.5 Implémentation de l'entrepôt de besoins

La phase d'implémentation de l'entrepôt de besoins consiste à implémenter le schéma en étoile de l'entrepôt dans un système de gestion de bases de données (SGBD) choisi. Comme nous souhaitons matérialiser le lien entre le schéma de l' \mathcal{EBF} et l'ontologie qui en définit le sens, nous avons opté pour un SGBD sémantique. Pour ce faire, nous avons effectué l'implémentation sur deux SGBD sémantiques : le SGBD commercial Oracle et le SGBD académique OntoDB développé au laboratoire LIAS de l'ISAE-ENSMA. Pour le cas d'Oracle, nous avons commencé l'implémentation en étendant le méta-schéma sémantique d'Oracle par le modèle de besoins que nous avons défini. L'articulation entre l'ontologie stockée et l'ontologie linguistique (qui est externe à Oracle) reste valide et se fait via des labels associés aux termes. Nous avons ensuite exploité l'algorithme ETL défini pour peupler l' \mathcal{EBF} par les instances des sources. Chaque opérateur est représenté par une requête Sparql. Par exemple, l'opérateur *Retrieve* est traduit comme suit :

```
Select ?instancesBesoin# Where ?instancesBesoin# rdf:type space:ClassREQ
```

Pour implémenter les opérateurs de raisonnement (identification, inférence et vérification de la consistance), nous avons mis en place des déclencheurs (*triggers*) qui implémentent la

base de règles de raisonnement. En suivant les mêmes étapes, nous avons réalisé la phase de conception physique en utilisant la BD sémantique OntoDB. OntoDB fournit son propre langage de définition et de manipulation des Ontologies qui est le langage OntoQL. Nous avons ainsi défini des requêtes OntoQL pour l'extension du méta-schéma d'OntoDB. Le schéma logique d'OntoDB suit une représentation horizontale dans laquelle chaque concept de l'ontologie est représenté par une table et chaque propriété est une colonne de la table. Le chargement des instances ontologiques dans ce schéma étendu s'est fait via le processus ETL. Cette implémentation a nécessité la traduction des opérateurs ETL en requêtes OntoQL de la même façon que pour la BDBO Oracle.

3.6 Analyses OLAP des besoins entreposés

La première étape pour l'extraction des informations sur les besoins à partir de l'entrepôt s'effectue en définissant des requêtes d'interrogation (OntoQL pour la BD OntoDB ou Sparql pour la BD sémantique Oracle). Nous avons également pensé à une analyse des besoins plus poussée, basée sur le concept de cube des entrepôts de besoins conventionnels. Notre solution consiste à utiliser QB4OLAP² qui est un vocabulaire supportant les analyses OLAP. Nous avons d'abord défini le cube multidimensionnel de besoins. Ensuite, nous avons défini des requêtes QB4OLAP correspondants à différents besoins décisionnels. Une série d'expérimentations a été menée pour évaluer la faisabilité de l'approche, en utilisant les besoins issus du document CMS, le benchmark ontologique LUBM³ et le SGBD sémantique Oracle (Djilani et al., 2016a). Nous avons évalué précisément les performances de l'algorithme ETL qui a indiqué un temps d'exécution polynomial. Une autre expérimentation a permis d'évaluer la pertinence du raisonnement pour l'obtention d'un entrepôt de qualité éliminant les besoins inconsistants (Djilani et al., 2016a). Un ensemble de relations ont été inférées par le processus de raisonnement pour chaque type de relation (*contain*, *refine*, *require*, *equal* et *conflict*). Pour certaines relations comme *require*, le taux de relation inférées dépasse les 50% de relations initiales. Le taux d'instances inférées pour chaque type de règles est également satisfaisant avoisinant les 30% (du nombre total d'instances) pour les règles d'identification, 50% pour les règles d'inférence et 20% pour les règles de vérification de la consistance. Un outil réalisant l'ensemble des phases de construction de l' \mathcal{EBF} présentées a également été fourni (<https://sites.google.com/site/abderrahmanekhiat/murgroom>).

4 Conclusion

Nous avons proposé dans cet article d'élever les solutions d'entreposage conventionnels, orientés données, vers une solution plus générique pour l'entreposage des besoins fonctionnels des utilisateurs. Nous avons suivi une approche de conception ontologique, qui a fait ses preuves dans les entrepôts conventionnels. Ce travail a nécessité un grand effort de réadaptation des solutions d'entreposage, suivant une approche holistique, concernant les aspects suivants : la conception du schéma de l'entrepôt, la modélisation multidimensionnelle du schéma, le mappings entre les schémas sources et le schéma cible, le processus ETL, l'implémentation de l'entrepôt et enfin l'analyse OLAP des besoins entreposés. Ce travail ouvre des perspectives

2. <https://www.w3.org/TR/vocab-data-cube-use-cases/>

3. <http://swat.cse.lehigh.edu/projects/lubm/>

qui concernent l'évaluation du passage à l'échelle de l'approche ainsi que la revisite des autres problématiques classiques des projets d'entreposage telles que la maintenance de l'entrepôt et la gestion de sa qualité.

Références

- Assawamekin et al. (2010). Ontology-based multiperspective requirements traceability framework. *Knowledge and Information Systems* 25(3), 493–522.
- Belaid, N., Y. Ait-Ameur, et J. F. Rainaud (2009). A semantic repository for geological modeling workflows. In *ICWS*, pp. 1030–1031. IEEE.
- Boukhari, I. (2014). *Intégration et exploitation de besoins en entreprise étendue fondées sur la sémantique*. Ph. D. thesis, ISAE-ENSMA Poitiers.
- Brottier, E., B. Baudry, Y. Traon, D. Touzet, et B. Nicolas (2007). Producing a global requirement model from multiple requirement specifications. In *EDOC*, pp. 390. IEEE.
- Djilani, Z., N. Berkani, et L. Bellatreche (2016a). Towards functional requirements analytics. In *ISOLA*, pp. 358–373. Springer.
- Djilani, Z., A. Khiat, S. Khouri, et L. Bellatreche (2016b). Murgroom : multi-site requirement reuse through graph and ontology matching. In *Proceedings of the 18th International Conference on Information Integration and Web-based Applications and Services*, pp. 160–169. ACM.
- Djilani, Z. et S. Khouri (2015). Understanding user requirements iceberg : Semantic based approach. In *Model and Data Engineering*, pp. 297–310. Springer.
- et al., J. T. (2012). Patentminer : topic-driven patent analysis and mining. In *ACM SIGKDD*, pp. 1366–1374.
- Giorgini, P. et al. (2003). Formal reasoning techniques for goal models. In *JODS*, pp. 1–20. Springer.
- Goknil, A. et al. (2011). Semantics of trace relations in requirements models for consistency checking and inferencing. *Software & Systems Modeling* 10(1), 31–54.
- Hannachi, L., N. Benblidia, F. Bentayeb, et O. Boussaid (2013). Social microblogging cube. In *DOLAP*, pp. 19–26.
- Khouri, S., L. Bellatreche, S. Jean, et A.-A. Y. (2014). Requirements driven data warehouse design : We can go further. In *ISOLA*, pp. 588–603.
- Liu, C. L. (2016). Cdnfre : Conflict detector in non-functional requirements evolution based on ontologies. *Computer Standards & Interfaces* 47, 62–76.
- López, O., M. A. Laguna, et F. J. G. Peñalvo (2002). Metamodeling for requirements reuse. In *WER*, pp. 76–90.
- Nguyen, T. H., J. Grundy, et M. Almorisy (2015). Integrating goal-oriented and use case-based requirements engineering : The missing link. In *MODELS*, pp. 328–337. IEEE.
- Rosa, M. L., H. A. Reijers, W. M. P. van der Aalst, R. M. Dijkman, J. Mendling, M. Dumas, et L. García-Bañuelos (2011). APROMORE : an advanced process model repository. *Expert Systems with Applications* 38(6), 7029–7040.

- Sun, Y. J., M. C. Barukh, B. Benatallah, et S. R. Beheshti (2015). Scalable saas-based process customization with casewalls. In *ICSOC*, pp. 218–233.
- Wache, H. et al. (2001). Ontology-based integration of information-a survey of existing approaches. In *ontologies and information sharing*, Volume 2001, pp. 108–117.
- Wieringa, R., E. Dubois, et H. S. Huyts (1997). Integrating semi-formal and formal requirements. In *CAiSE'97*, pp. 19–32.

Summary

In recent years, data warehouse (DW) and data analysis technology has become mature. This maturity has prompted a significant number of companies to adopt this technology. Analyzing this situation, we notice that DWs have often been *associated with data*. However, in the one hand, other important entities (like models, services, processes, etc) used by companies have the same interest to be analyzed. On the other hand, many efforts have been made to integrate these entities into repositories, but without the analysis dimension of DWs. Combining the two solutions proposed by repositories and DWs would be an important asset for both companies and researchers. In this study, we propose a complete approach for developing functional requirements warehouses, which can be considered as a pre-condition for designing any type of warehouse or repository.

