Osons tout Persister dans un Entrepôt: Données et Modèles

Selma Khouri **, Ladjel Bellatreche*

** Ecole nationale Supérieure d'Informatique ESI, Algérie
LISI/ENSMA, Université de Poitiers, France
s_khouri@esi.dz
*LISI/ENSMA, Université de Poitiers, France
bellatreche@ensma.fr

Résumé. La structure de stockage actuelle des entrepôts de données matérialise des données conformément à une modèle logique défini par les concepteurs. Récemment, plusieurs travaux ont montré l'intérêt de la modélisation conceptuelle dans le contexte des entrepôts de données du fait qu'elle offre une abstraction du domaine étudié. Sa disponibilité facilite l'interrogation de l'entrepôt de données, car il est plus riche que le modèle logique. Les modèles conceptuel et logique de l'entrepôt sont définis à partir de deux composantes : les sources de données et les besoins des utilisateurs. L'utilisation des besoins utilisateurs est actuellement reconnue comme une étape indispensable pour la réussite des projets d'entreposage. L'utilisation des besoins se fait sentir à différentes phases du cycle de vie de l'entrepôt (optimisation, personnalisation, recommandations, traçabilité, etc.). Nous remarquons cependant qu'aucune trace du modèle conceptuel ni du modèle des besoins n'est sauvegardée dans l'entrepôt final. Pour remédier à ces limites, nous proposons dans cet article une piste de réflexion sur la proposition d'une nouvelle structure de stockage d'entrepôt permettant de représenter d'une manière persistante ces trois modèles : le modèle conceptuel, le modèle des besoins et le modèle logique.

1 Introduction

La conception d'un entrepôt de données (ED) permet de générer le modèle logique et physique de l'entrepôt. La structure de stockage actuelle d'un ED représente le modèle logique de l'entrepôt, généralement sous forme d'un schéma relationnel, et stocke les données conformément à ce modèle. La phase de modélisation conceptuelle a d'abord été ignorée lors de la conception des projets d'EDs. Plusieurs études ont par la suite reconnue cette phase comme une étape importante offrant une vue abstraite du domaine étudié, et facilitant la communication entre les concepteurs et les utilisateurs de l'entrepôt (Gam et Salinesi, 2006). Plusieurs méthodes de modélisation conceptuelle d'ED ont été proposées. Le modèle conceptuel est également utilisé lors de la phase d'exploitation de l'entrepôt où sa disponibilité est nécessaire à l'utilisateur final pour comprendre le schéma physique de l'entrepôt et faciliter son interrogation. Nous remarquons cependant que, seul le modèle logique est représenté dans une structure classique d'entrepôt, et qu'aucune trace du modèle conceptuel n'est sauvegardée. Ce schéma

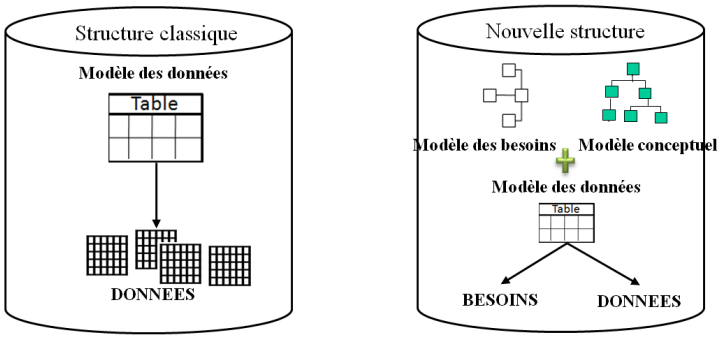

FIG. 1 – *La structure classique d'un ED et la nouvelle structure de stockage proposée.*

logique de l'ED n'est pas aussi expressif que son schéma conceptuel. Il renferme très souvent des décisions d'optimisation et d'implémentation et plusieurs constructeurs conceptuels perdent leurs sémantiques lors de la traduction.

La définition du modèle final de l'entrepôt repose sur deux composantes essentielles qui sont : les sources de données et les besoins des utilisateurs. Dans les premiers travaux sur la conception d'ED, les besoins des utilisateurs n'ont pas été pris en compte. Actuellement, ils reconnus par les praticiens du domaine et par la communauté de recherche, comme un facteur clé déterminant le succès ou l'échec du projet d'entreposage (Rizzi et al., 2006). Le cycle de développement de l'entrepôt inclut une première étape d'analyse des besoins (Golfarelli et Rizzi, 2009). Différentes méthodes suivent cette tendance et proposent des méthodes de conception orientées besoins ou mixtes (sources et besoins). Ces besoins des utilisateurs seront exploités à différentes phases du cycle de vie de l'ED : (1) les besoins et requêtes des utilisateurs sont utilisés au niveau physique où certains SGBDs (comme Oracle ou SQL Server) stockent leurs requêtes au niveau de la base pour des fins d'optimisation. Une requête peut être vue comme une instance d'un besoin. (2) les besoins sont utilisés pour définir le schéma de stockage le plus adéquat pour l'entrepôt (relationnel, multidimensionnel, hybride, etc.), (3) Les préférences des utilisateurs, qui représentent des besoins non fonctionnels, sont exploitées pour la personnalisation des requêtes et pour les recommandations, (4) Les besoins sont également tracés pour le suivi de l'évolution de l'application décisionnelle, (5) la qualité de l'application finale est mesurée en fonction de sa capacité à répondre aux besoins et buts identifiés. Nous remarquons cependant, malgré la présence et l'utilisation des besoins dans ces différentes phases d'exploitation, qu'aucune trace des besoins n'est sauvegardée au sein de l'entrepôt. Ce manque nécessite donc un double travail de collecte et d'analyse des besoins pouvant engendrer une perte de temps considérable.

Pour apporter une solution aux deux manques identifiés : l'absence d'une vue conceptuelle des données et l'absence d'une représentation des besoins dans l'entrepôt final, nous proposons une nouvelle structure de stockage d'ED, étendant et complétant la structure classique, et qui permettra de stocker l'ensemble de ces modèles (Figure 1).

Le modèle de l'entrepôt dans cette nouvelle structure présente un socle commun où cohabitent plusieurs sous-modèles : le modèle logique de données habituel, le modèle des besoins et le modèle conceptuel des données donnant une sémantique aux deux modèles précédents.

Nous avons conçu ce modèle d'entrepôt de manière à ce qu'il soit extensible afin de pouvoir y représenter d'autres types de modèles. La principale innovation de cette structure de stockage, en plus de représenter le modèle conceptuel, est qu'elle revalorise les besoins et leur redonne une place centrale dans l'entrepôt au même titre que les données. Nous estimons, que le modèle des besoins est une composante essentielle du modèle de l'entrepôt, pouvant être exploité tout au long du cycle de vie de l'entrepôt de manière verticale au niveau de chaque phase du cycle de vie, et de manière horizontale d'une phase à une autre en assurant la traçabilité des besoins. Ce papier est structuré en cinq sections : la section 2 présente un état de l'art des méthodes de conception d'EDs. Nous y montrons l'importance du modèle conceptuel et les besoins des utilisateurs. Dans la section 3, nous nous interrogeons sur l'exploitation des besoins à travers le cycle de vie de l'ED, et l'utilité d'avoir une représentation persistante des besoins au sein de l'entrepôt. La section 4 propose une structure de stockage d'ED supportant les trois modèles cités. La section 5 présente la conclusion et quelques perspectives à explorer.

2 Etat de l'art

En explorant les principaux travaux de conception d'EDs, nous remarquons une première tendance proposant de générer directement le modèle logique de l'entrepôt généralement à partir des sources de données (M. et vom Ende, 1999), (Moody et Kortnik, 2000), (Song et al., 2007). D'autres travaux ont suivis reconnaissant l'importance d'une phase de modélisation conceptuelle (Golfarelli et Rizzi, 1998), (Luján-Mora et Trujillo, 2003) (Prat et al., 2006). Cette phase permet de fournir un modèle sémantiquement riche (comparé aux modèles logique et physique), présentant une abstraction de la réalité qui est proche de la façon de penser des utilisateurs et fournissant une documentation essentielle tout au long du processus de développement de l'ED. (Franconi et al., 200O) souligne que les EDs et les applications OLAP, par la présence des agrégations multidimensionnelles, requièrent des formalismes de modélisation conceptuelle suffisamment expressifs. Ces méthodes citées génèrent le schéma conceptuel ou logique de l'ED à partir de schémas relationnels des sources comme (Jensen et al., 2004), de schémas conceptuels des sources (M. et vom Ende, 1999), (Moody et Kortnik, 2000), (Song et al., 2007), ou à partir d'un schéma ontologique des sources (Romero et al., 2009), (Nebot et al., 2009), (Khouri et Bellatreche, 2010a). L'utilisation des ontologies a été identifiée comme solution efficace à la problématique d'intégration des sources.

Dans ces différents travaux, le modèle conceptuel ou logique de l'entrepôt est généré à partir des sources de données et des besoins des utilisateurs. Un ED, par définition, doit être construit à partir des sources de données. Une analyse centrée sources est par conséquent indispensable. Les besoins des utilisateurs ont d'abord été ignorés dans les premiers travaux de conception. Il est actuellement reconnu que le cycle de développement d'un ED implique une première phase d'analyse des besoins (Golfarelli et Rizzi, 2009). Le domaine qui s'intéresse à l'analyse des besoins est le domaine de l'ingénierie des besoins. L'ingénierie des besoins est définie comme le processus permettant d'identifier les attentes des utilisateurs d'un système et de les documenter de manière à faciliter leur analyse, leur communication et leur implémentation (Nuseibeh et Easterbrook, 2000). Dans le cas d'un projet d'ED, plusieurs méthodes de conception orientées besoin (List et al., 2000), (Mazon et al., 2008) ou mixtes (Giorgini et al., 2005), (Bonifati et al., 2001) ont été proposées. L'expérience a montré que même après implémentation de l'entrepôt de données, un retour vers la phase d'identification des besoins est

souvent effectué (Ballard et al., 1998). Les besoins identifiés en ED peuvent être fonctionnels ou non-fonctionnels. Un besoin non fonctionnel est défini comme un attribut ou une contrainte du système comme la flexibilité, performance, sécurité, etc (Glinz, 2007).

L'analyse des besoins dans un projet d'EDs diffère selon l'objet analysé. On retrouve une analyse orientée processus, orientée utilisateur ou orientée but. L'analyse orientée processus (List et al., 2000), (Bruckner et al., 2001), (Schiefer et al., 2002) analyse les besoins en identifiant les processus métiers de l'organisation. L'analyse orientée utilisateurs (Winter et Strauch, 2003) s'intéresse à identifier les utilisateurs cibles et à spécifier leurs besoins individuels pour les intégrer dans un modèle de besoins unifié. L'analyse orientée but (Bonifati et al., 2001), (Giorgini et al., 2005), (Gam et Salinesi, 2006), (Mazon et al., 2008), (Kumar et al., 2010) identifie les buts et objectifs de l'organisation qui guident les décisions de différents niveaux de l'organisation. Il a été reconnu par plusieurs auteurs que l'analyse orientée but offre une meilleure définition des besoins des utilisateurs (Mylopoulos et al., 2001), (Bonifati et al., 2001), (Rolland, 2005). Identifier les objectifs et buts des utilisateurs en début de projet est une étape cruciale dans le processus de développement (Nuseibeh et Easterbrook, 2000), spécialement dans les projets d'applications décisionnelles qui visent à analyser l'activité d'une organisation et où la réalisation des buts est un indicateur important de cette activité (Stefanov et List, 2006).

La modélisation des besoins dans les travaux de conception d'ED s'effectue sous différentes formes : en utilisant des modèles spécifiques proposés par les auteurs mêmes comme (Gam et Salinesi, 2006) avec le modèle MAP, en utilisant les modèles fournis par le framework i* [1] comme (Giorgini et al., 2005), ou en utilisant une modélisation objet comme (List et al., 2000), (Mazon et al., 2008). Nous avons proposé dans (Khouri et Bellatreche, 2010a) de modéliser les besoins des utilisateurs d'un ED au niveau ontologique. L'utilisation des ontologies présente de nombreux avantages pour : (1) la spécification des besoins, (2) la compréhension du domaine, (3) les capacités de raisonnement des ontologies peuvent être exploitées pour corriger les inconsistances et erreurs de modélisation et pour inférer de nouveaux faits sur les besoins, (4) dans les domaines faisant intervenir plusieurs domaines, la modularité des ontologies permet la réutilisabilité et l'extraction des modules nécessaires à partir de différentes ontologies de domaine, utilisés comme support au modèle des besoins.

3 Exploitation des besoins durant le cycle de vie d'un ED

Nous défendons l'idée que le modèle des besoins doit être représenté au niveau du modèle de l'entrepôt au même titre que le modèle des données. Nous montrons dans ce qui suit les différents avantages de la persistance des besoins tout au long du cycle de vie de l'entrepôt pendant ses phases de conception, d'exploitation et de maintenance (Figure 2).

3.1 Les besoins dans la conception de l'entrepôt

L'exploitation des besoins des utilisateurs permet dans un premier temps de générer le modèle conceptuel multidimensionnel de l'ED répondant aux exigences des décideurs et utili-

1. i* est un framework qui repose sur les notions d'agent et de but dans les différentes phases de développement d'un système et permet d'identifier les utilisateurs du système et les modéliser en acteurs sociaux dépendants les uns des autres à travers les buts à réaliser, les tâches à effectuer et les ressources à utiliser.

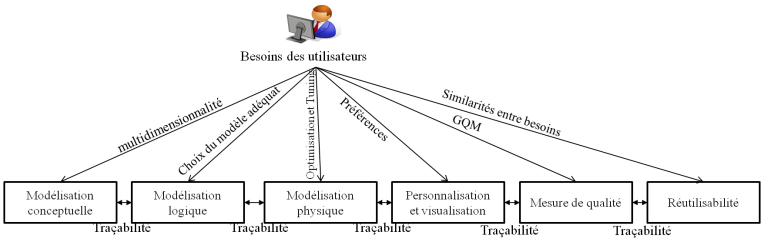

FIG. 2 – *Utilisation des besoins en phase de conception, exploitation et maintenance de l'ED.*

sateurs. L'exploitation des besoins se fait également sentir lors du choix du schéma de stockage logique. Ce schéma peut suivre différentes représentations : relationnelle, multidimensionnelle, hybride, et de plus en plus une représentation Nosql. La représentation la plus utilisée est la représentation relationnelle sous forme d'un schéma en étoile, en flocon ou en constellation. La représentation multidimensionnelle implémente les cubes de l'entrepôt sous forme d'un tableau multidimensionnel. Une représentation hybride représente un schéma mixant les deux représentations précédentes. On parle actuellement de plus en plus du mouvement NoSql (Not Only SQL) lancé en 2007 par les deux compagnies Amazon et Google (Leavitt, 2010). Plusieurs organisations collectant et manipulant de grandes quantités de données, spécialement celles du Web 2.0 (Google, Amazon, EBay, Facebook, etc), se tournent actuellement vers les modèles de données NoSql consistant en des modèles de stockage plus flexibles (où le nombre d'attributs est variable, où les valeurs nulles sont tolérées et où plusieurs champs sont optionnels), comme les modèles clé-valeur, les modèles orientés document ou les modèles orienté colonnes (Leavitt, 2010). D'autres schémas de stockage sont discutés dans (Grabova et al., 2010). Le choix de la structure de représentation logique la plus appropriée dépend fortement des besoins de l'application et des requêtes des utilisateurs (une requête est vue comme une instance d'un besoin). Les modèles NoSql sont ainsi plus adéquats lorsque les requêtes des utilisateurs se présentent en très grand nombre sous forme de requêtes simples impliquant de simples opérations, favorisant l'accès aux données en lecture plutôt qu'en écriture, identifiées généralement par clé primaire, et n'impliquant pas beaucoup de jointures. Les modèles relationnels sont utilisés lorsque les requêtes sont plus complexes. Les modèles de stockage multidimensionnels sont utilisés pour favoriser l'accès rapide aux données.

3.2 Optimisation des requêtes

L'exploitation des besoins et requêtes des utilisateurs apparaît également au niveau physique où plusieurs SGBDs (comme Oracle ou Sql Server) stockent leurs requêtes pour des fins d'optimisation. Les requêtes (et parfois même le plan d'exécution de la requête) sont stockées généralement sous forme de chaînes de caractères formant un journal de requêtes sauvegardant l'historique des besoins des utilisateurs de l'application. De la même manière, le stockage des besoins peut se faire dès le début de la conception afin de prévoir certaines structures d'optimisation adéquates.

La disponibilité des besoins et requêtes et leur stockage lors de la conception et l'exploitation de l'entrepôt permet également de constituer une base de requêtes servant comme noyau pour un benchmark opérationnel. Ceci est d'autant intéressant dans les domaines manquant de benchmarks. Notons par exemple qu'il n'existe actuellement aucun benchmark dédié aux entrepôts sémantiques (comme les web warehouse). L'exploitation d'un entrepôt ontologique stockant le modèle des besoins permet de constituer un tel benchmark.

3.3 Personnalisation et recommandation

Les préférences des utilisateurs sont définies comme des besoins non fonctionnels comme la sécurité ou la qualité (Tapucu, 2010). Une des principales tâches dans un processus d'aide à la décision est d'analyser les préférences des utilisateurs de manières à déduire des recommandations finales exploitables par le décideur (Tapucu, 2010). Modéliser les besoins non-fonctionnels comme les préférences explicitement au sein du modèle de l'entrepôt fournit une base permettant de capturer ces préférences et de les exploiter pour mieux répondre aux requêtes des utilisateurs. La personnalisation repose sur la notion de préférence. Elle représente le processus permettant d'adapter et de filtrer les informations en fonction des préférences individuelles d'un utilisateur. La personnalisation des requêtes permet également d'améliorer les requêtes des utilisateurs en les enrichissant avec leurs préférences, afin de présenter des réponses plus pertinentes et des résultats personnalisés. Cette personnalisation peut même se faire à travers des interfaces customisées offrant ainsi une visualisation des données adéquate à chaque profil d'utilisateur. (Bellatreche et al., 2005) analyse également l'impact des préférences sur la conception physique de l'entrepôt (partitionnement des données, sélection des vues matérialisées et des index). Le modèle des besoins peut stocker les préférences des utilisateurs en sauvegardant les liens entre les besoins non fonctionnels identifiés et les acteurs interagissant avec ces besoins. Avoir ce modèle des besoins dans l'entrepôt permet de constituer une base des préférences des utilisateurs s'enrichissant tout au long de l'exploitation de l'entrepôt, et servant comme support pour la recommandation et pour la personnalisation.

3.4 Mesure de qualité

La gestion de la qualité en développement de logiciel commence par la compréhension des besoins des utilisateurs (Aybuke et Claes, 2005). La qualité de systèmes logiciels se mesure en fonction de leur capacité à répondre aux buts et objectifs fixés. La méthode GQM (Goal Question Metric) par exemple, est une technique permettant de fournir des métriques afin de mesurer des programmes. GQM repose essentiellement sur les buts identifiés d'une organisation pour identifier ces métriques.

3.5 Réutilisabilité

Lorsque des similarités entre les besoins d'une application existante et une nouvelle application sont identifiées, cette information permet d'identifier les composants réutilisables à exploiter dans la nouvelle application pour implémenter les besoins similaires (Aybuke et Claes, 2005). Cette réutilisabilité peut se faire au niveau des besoins ou au niveau des composants. Le stockage des besoins facilite cette réutilisabilité en identifiant pour chaque besoin, les données associés et les composants du système associés.

3.6 Traçabilité des besoins

La traçabilité est définie par la possibilité de décrire et de suivre la vie d'un besoin dès son origine, son développement et son spécification jusqu'à son implémentation et son utilisation (Nuseibeh et Easterbrook, 2000). La traçabilité implique la connaissance des liens entre les besoins et de leurs interdépendances. La traçabilité permet notamment : d'identifier le rôle de chaque objet créé, d'assurer la propagation des changements, d'évaluer l'impact de toute modification ou mise à jour sur les données et sur les accès à l'entrepôt (Aybuke et Claes, 2005). L'exploitation et la maintenance de l'entrepôt repose donc sur la traçabilité des besoins. Le stockage des besoins dans l'entrepôt permet d'assurer cette traçabilité des besoins depuis leur identification.

4 Notre proposition

Nous proposons une nouvelle structure de stockage d'ED dont le modèle présente une base commune stockant plusieurs modèles : le modèle conceptuel définissant la sémantique de l'univers du discours, le modèle des besoins fournissant une vue persistante des besoins et requêtes des utilisateurs et le modèle logique des données. Nous analysons dans la section suivante les composantes de chaque modèle, et décrivons comment ces modèles peuvent cohabiter au sein de cette nouvelle structure.

4.1 Le modèle conceptuel des données

Les méthodes de conception présentes dans l'état de l'art (section 2) peuvent être utilisées afin de définir le modèle conceptuel de l'entrepôt. Nous avons proposé dans (Khouri et Bellatreche, 2010a) une méthode de conception permettant de définir le schéma conceptuel multidimensionnel d'un ED (Figure 3). Cette méthode repose sur une ontologie de domaine décrite en OWL utilisée comme schéma d'intégration des sources de données en entrée. Partant du constat qu'un ED peut être vu comme un système d'intégration matérialisé par des concepts multidimensionnels, nous avons rapidement remarqué que les ontologies sont considérées comme la clé de l'automatisation du processus d'intégration (Bellatreche et al., 2006). Cette intégration suppose l'existence d'une ontologie de domaine, et de sources ontologiques référençant cette ontologie. Cette référence peut se faire à postériori à travers des algorithmes de matching, ou à priori (Bellatreche et al., 2006).

Nous avons par ailleurs constaté que les ontologies ont largement contribué dans le domaine de l'ingénierie des besoins pour faciliter l'expression et la formalisation des besoins de différentes formes dans diverses applications. Les ontologies permettent notamment de (Calero et al., 2006) : faciliter l'expression des besoins, représenter le modèle des besoins de manière formelle, d'exploiter des mécanismes de raisonnement permettant d'identifier les besoins contradictoires ou conflictuels et d'inférer de nouveaux faits sur ces besoins. Nous avons proposé de converger ces deux domaines (ingénierie des besoins et ontologies) dans le cadre spécifique d'un entrepôt de données, et nous avons représenté les besoins décisionnels d'un entrepôt en utilisant les propriétés et concepts ontologiques de l'ontologie de domaine intégrant les sources de données. La projection des besoins sur l'ontologie intégrante permet de définir une ontologie locale à l'ED considéré comme son modèle conceptuel. Une ontologie peut être

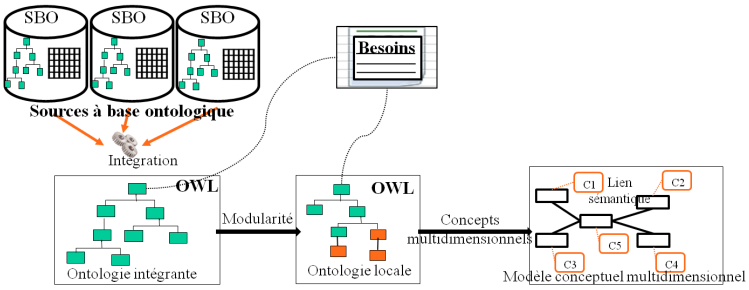

FIG. 3 – *méthode de conception ontologique proposée.*

vue comme un modèle conceptuel ayant des capacités de raisonnement. L'ontologie locale est extensible afin de permettre la représentation de nouveaux besoins identifiés. Elle est annotée par les concepts multidimensionnels. Nous avons validé cette méthode par un outil de génie logiciel implémentant toutes les étapes de cette conception (Khouri et Bellatreche, 2010b).

4.2 Le modèle des besoins

Nous avons opté, pour définir les besoins des utilisateurs, pour une représentation orientée but. L'analyse orientée but repose sur l'identification des buts et objectifs d'une organisation pour définir les données à modéliser. Elle permet de valider les besoins collectés en identifiant les buts conflictuels et de montrer les différentes alternatives de modélisation permettant d'atteindre un but donné, ainsi que les différentes influences entre ces alternatives (Giorgini et al., 2003). Ces alternatives peuvent être exploitées afin de mieux cerner les données pertinentes à représenter au niveau de l'entrepôt. A partir de la lecture des différentes modélisations des buts proposées dans la littérature (Bonifati et al., 2001), (Giorgini et al., 2005), (Gam et Salinesi, 2006), (Mazon et al., 2008), (Kumar et al., 2010), (Stefanov et List, 2006), nous retenons les principales notions représentées dans le modèle de la figure 4.

Prenons l'exemple du but suivant : " *Améliorer la qualité d'enseignement des cours dans les filières A et B selon le directeur des études*". Cette formulation représente la manière dont les utilisateurs expriment généralement leurs buts. Nous considérons qu'un but est décrit par les attributs suivants : objectif (*Améliorer*), une priorité (on distingue généralement un but obligatoire d'un but optionnel) et un label permettant d'indiquer si le but est satisfait, insatisfait, indéterminé ou conflictuel. L'attribut priorité peut aider à gérer les conflits entre les buts. Un but possède quatre principales coordonnées : un résultat à analyser (*qualité d'enseignement*), un paramètre d'analyse (*cours* et *filière*), une métrique permettant de mesurer le but (*le nombre d'étudiants inscrits assistant au cours*) et un acteur (*directeur des études*). L'acteur représente tout individu concerné par ce but, ou qui interagit avec le système à travers ce but. Il peut être : une ou plusieurs personnes internes ou externes à l'organisation, un service de l'organisation ou même un système logiciel. Cette entité sera utile par la suite lors de la phase de personnalisation. Les relations entre les buts sont de trois types : les relations ET/OU permettant de décomposer un but en sous-buts, et les relations d'influence positive, négative ou ambiguë. Différentes méthodes ont été proposées permettant : l'acquisition de ces buts, l'identification

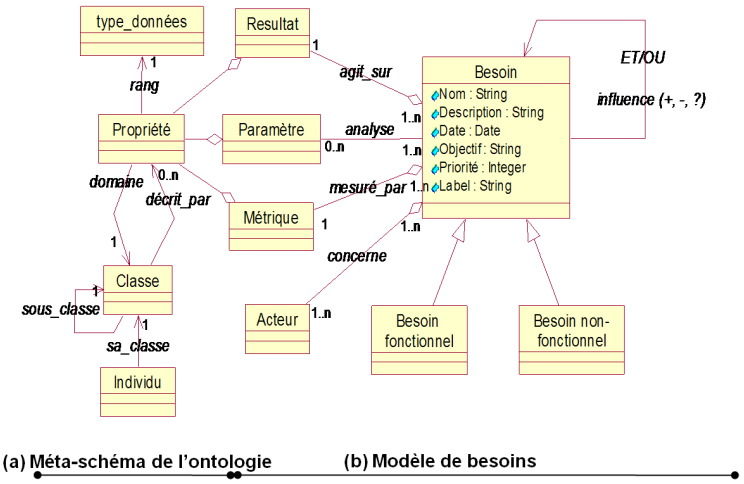

FIG. 4 – *Modèle des besoins proposé défini au niveau ontologique.*

des buts à partir de scénarios, la découverte des relations ET/OU et des relations d'influence entre les buts (Rolland, 2005).

Afin de représenter ce modèle de besoins au niveau conceptuel ou ontologique, et de profiter des avantages de l'ontologie en terme de formalisation et de raisonnement, nous proposons d'étendre le modèle ontologique par le modèle des besoins. Les coordonnées de chaque besoin s'expriment en utilisant les propriétés ontologiques comme illustré en figure 4. Ces besoins sont exprimés sur les propriétés ontologiques, pour permettre au concepteur de choisir uniquement les propriétés pertinentes d'une classe pour représenter ce but. Il est ainsi possible de connaître pour chaque besoin identifié, l'ensemble des concepts et propriétés nécessaires à sa réalisation, qui seront représentés dans l'entrepôt.

4.3 Le modèle logique

Le modèle conceptuel que nous avons défini en section 4.1 est représenté par une ontologie. Différentes méthodes ont été proposées dans la littérature permettant la traduction d'une ontologie décrite selon un format spécifique (PLIB, RDF, OWL) en un schéma logique, généralement relationnel ou relationnel-objet (Pan et Heflin, 2003), (Gali et al., 2004), (Vysniauskas et Nemuraite, 2006), (Narayanan et al., 2007). La traduction d'une ontologie en un schéma relationnel se fait selon trois représentations possibles : verticale, binaire ou horizontale (Bellatreche et al., 2010). Une représentation *verticale* stocke les données en une table unique à trois colonnes (sujet, prédicat, objet). Cette représentation est adaptée pour les ontologies RDF. Dans une représentation *binaire*, les classes et les propriétés ontologiques sont stockées dans des tables de structures différentes. Une représentation *horizontale* associe à chaque classe de l'ontologie une table ayant une colonne pour chaque propriété de la classe. Nous avons établi dans le cadre de la thèse de (Fankam, 2009) effectuée au laboratoire LISI des règles de correspondances permettant de traduire les constructeurs d'une ontologie OWL (classe, propriété,

restriction) selon les deux représentations relationnelles (binaire et horizontale) que nous résumons en ces points :

- Approche horizontale
  - Classe : chaque classe correspond à une table. Cette table contient un champ représentant la clé primaire de type entier et un champ pour chaque propriété applicable à cette classe.
  - Propriété de type simple (Datatype property) : toute propriété de type simple monovaluée i.e de cardinalité (0 :1 ou 1 :1) est représentée comme un champ dans la table correspondant à la classe. Toute propriété collection (multi-valuée) est représentée par une table d'association. Une clé étrangère référençant la table correspondant à la classe définissant la propriété est crée. Un second champ contenant chaque valeur de collection est crée. Chaque type de données XSD supporté par le formalisme OWL possède son type de données SQL correspondant (Exemple. Short devient SMALLINT).
  - Propriété de type objet (Object property) : chaque propriété objet mono-valuée est représentée dans la table correspondant à la classe qui la définit (classe domaine) comme une clé étrangère référençant la clé primaire de la table correspondant à la classe référencée (classe rang). Les propriétés objet de type collection présentant une association (1 :N) sont représentées dans la table correspondant à la classe référencée par une clé étrangère référençant la clé primaire de la table référençante (classe domaine). Les propriétés objet de type collection présentant une association (N :M) sont représentées par une nouvelle table d'association comportant les champs suivants : une clé étrangère référençant la table correspondant à la classe domaine de la propriété, un champ contenant l'identifiant de l'instance cible et un champ spécifiant la table qui contient cette instance. La clé primaire de la table est constituée des deux premiers champs.
- Approche binaire
  - Classe : chaque classe correspond à une table contenant un seul champ représentant une clé primaire de type entier.
  - Propriété de type simple : chaque propriété de type simple (monovaluée et multivaluée) est représentée par une table d'association dont les champs sont : une colonne de la clé primaire identifiant les individus de la classe domaine, une deuxième colonne contenant les valeurs de propriété et une troisième colonne de type chaîne de caractère permettant de référencer la table de l'individu (dans la hiérarchie des classes).
  - Propriété de type objet : pour chaque propriété de type objet (monovaluée et multivaluée), une table d'association est crée. Les champs de cette table sont : une colonne pour la clé primaire de la table représentant les identifiants des instances de la classe source (domaine), une colonne représentant les identifiants de la classe cible (co-domaine), deux colonnes de type chaîne de caractères spécifiant la table des instances source et cible.

Les caractéristiques des propriétés (symétrie, transitivité), pour les deux approches binaire et horizontale, s'effectue par la création de triggers. Les différentes restrictions OWL sont traduites sous forme de vues. Le choix de la représentation (binaire ou horizontale) se fait en fonction des besoins de l'application et du type de requêtes utilisées (Dehainsala et al., 2007).

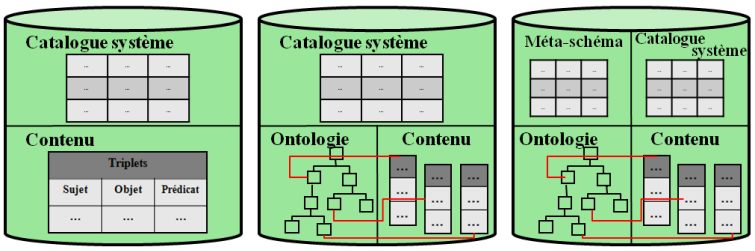

FIG. 5 – *Les architectures type I, type II, type III d'une BDBO.*

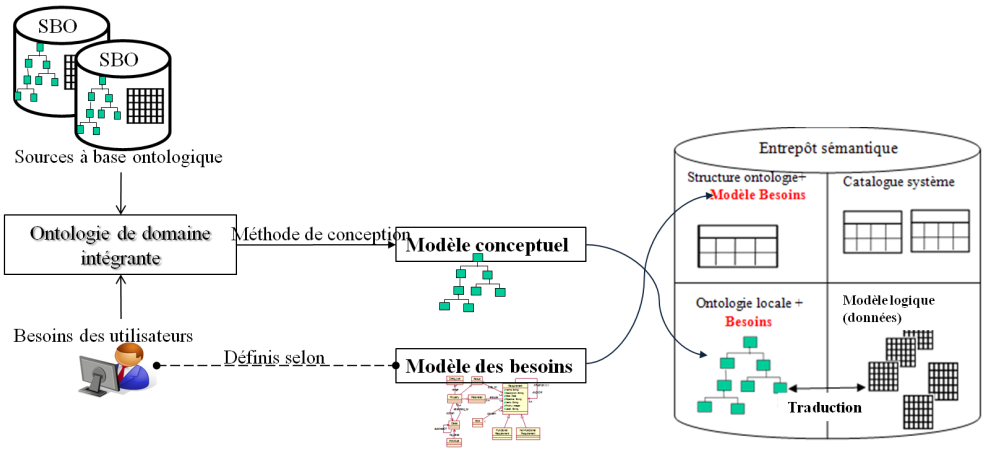

FIG. 6 – *La structure de stockage proposée et persistance des modèles conceptuel, besoins et logique.*

4.4 Persistance des modèles dans la nouvelle structure de stockage

L'architecture de base de données (BD) permettant de stocker les données et l'ontologie (le modèle conceptuel étendu pas le modèle des besoins) qui définit la sémantique des données est appelée Base de Données à Base Ontologique. Les BDBOs ont suscité l'intérêt des deux communautés industrielle (Oracle et IBM) et académique (OntoDB (Fankam, 2009)). La représentation de l'ontologie dans une BD donne lieu à plusieurs types d'architectures de BDBOs que nous classons comme suit (Bellatreche et al., 2010) : type I, type II et type III (Figure 5).

Dans les BDBOs de *type I*, l'ontologie et les données à base ontologique (DBOs) associées sont stockées dans un même schéma. Cette architecture facilite les mises à jour des instances et propriétés. Ses performances sont cependant dégradées à cause des nombreuses opérations d'auto-jointure effectuées sur une table unique (Exp. Jena2 (Carroll et al., 2004)). Dans une architecture de *type II*, l'ontologie et les données associées sont stockées dans deux schémas différents : un schéma pour les DBOs, et un autre pour l'ontologie (Exp. RDF Suite (Alexaki et al., 2001)). L'architecture de *type III* étend celle de type II en définissant un schéma sup-

plémentaire appelé méta-schéma (Exp.OntoDB (Fankam, 2009)). L'ontologie constitue ainsi une instance de ce méta-schéma. La présence du méta-schéma offre une flexibilité de l'ontologie et permet : (i) l'évolution du formalisme d'ontologie utilisé, (ii) un accès générique aux ontologies ainsi qu'aux données et (iii) le stockage de différents formalismes d'ontologies.

Nous optons pour une architecture de type III pour la représentation de la structure d'entrepôt que nous proposons. La présence du méta-schéma offre une grande flexibilité et permet son extension par le modèle des besoins. Il est possible d'étendre davantage ce schéma par de nouveaux modèles comme par exemple un modèle de préférences détaillé. La figure 6 présente la structure de stockage finale de l'entrepôt représentant les modèles définis. Dans cette structure, l'ED est constitué du modèle logique (relationnel) de données, de son méta-modèle (catalogue système), du schéma de l'ontologie locale représentant le modèle conceptuel de l'ED (obtenue par application de la méthode ontologique présentée en section 4.1) et du méta-schéma de l'ontologie étendu par le modèle de besoins.

5 Conclusion

La modélisation conceptuelle d'un ED représente une étape indispensable du développement d'un projet d'entreposage. Le modèle conceptuel d'un ED est généré à partir d'une analyse des sources de données et des besoins des utilisateurs. Les besoins des utilisateurs représentent actuellement une composante essentielle pour la conception d'un ED. L'utilisation de ces besoins ne s'arrête pas à la phase de conception, mais se fait sentir en différentes phases d'exploitation et de maintenance de l'entrepôt. Aucune trace des besoins n'est cependant sauvegardée au niveau des EDs une fois la phase de conception achevée. Après avoir identifié ce manque, nous avons pensé à proposer une nouvelle structure de stockage d'ED permettant de représenter plusieurs modèles de manière persistante, dont le modèle des besoins. Nous avons commencé par identifier l'importance accordée par la communauté de recherche aux deux modèles : le modèle conceptuel et le modèle des besoins, et nous avons analysé l'utilisation des besoins à travers différentes étapes du cycle de vie de l'entrepôt. La structure de stockage proposée repose sur un modèle d'entrepôt flexible et extensible, faisant cohabiter plusieurs modèles comme : le modèle des besoins, le modèle conceptuel et le modèle logique (le seul modèle représenté dans une structure classique d'ED). Nous avons exploité nos différents travaux, ainsi que les travaux proposés dans la littérature afin de définir chacun de ces modèles, et nous avons défini une nouvelle structure de stockage d'ED représentant ces modèles de manière persistante.

Comme perspectives, nous envisageons d'étudier l'effet de stockage des besoins sur les différentes phases d'exploitation de l'ED principalement la phase d'optimisation et de personnalisation, et d'étudier l'évolution de l'ontologie et sa maintenance au sein de l'entrepôt. Nous prévoyons également de valider cette nouvelle structure sur un cas d'étude réel, où nous exploiterons la BDBO de type III (OntoDB) développée au laboratoire LISI/ENSMA.

Références

Alexaki, S., V. Christophides, G. Karvounarakis, D. Plexousakis, et K. Tolle (2001). The ics-forth rdfsuite: Managing voluminous rdf description bases. *International Workshop on the*

Semantic Web, SemWeb, 1–13.

Aybuke, A. et W. Claes (2005). *Engineering and Managing Software Requirements*. Springer.

Ballard, C., D. Herreman, D. Schau, R. Bell, E. Kim, et A. Valencic (1998). Data modeling techniques for data warehousing. International technical support organization, ibm, RNTI (http://www.redbooks.ibm.com).

Bellatreche, L., Y. A. Ameur, et C. Chakroun (2010). A design methodology of ontology based database applications. *Logic Journal of the IGPL, Oxford University Press*, 679–696.

Bellatreche, L., A. Giacometti, P. Marcel, H. Mouloudi, et D. Laurent (2005). A personalization framework for olap queries. *Proceedings of the 8th ACM international workshop on Data warehousing and OLAP DOLAP05*, 9–18.

Bellatreche, L., D. N. Xuan, G. Pierra, et H. Dehainsala (2006). Contribution of ontology-based data modeling to automatic integration of electronic catalogues within engineering databases. *Computers inIndustry Journal Elsevier 57*, 711–724.

Bonifati, A., F. Cattaneo, S. Ceri, A. Fuggetta, et S. Paraboschi (2001). Designing data marts for data warehouses. *ACM Transactions on Software Engineering and Methodology 10*, 452–483.

Bruckner, R., B. List, et J. Schiefer (2001). developing requirements for data warehouse systems with use case. *Seventh Americas Conference on Information Systems*.

Calero, C., F. Ruiz, et M. Piattini (2006). *Ontologies for Software Engineering and Software Technology*. Berlin Heidelberg: Springer Verlag.

Carroll, J., I. Dickinson, C. Dollin, D. R. A. Seaborne, et K. Wilkinson (2004). Jena: implementing the semantic web recommendations. *WWW04*, 74–83.

Dehainsala, H., G. Pierra, et L. Bellatreche (2007). Ontodb : An ontology-based database for data intensive applications. *12th International Conference on data-base systems for advanced applications DASFAA*, 497–508.

Fankam, C. (2009). *OntoDB2 : un système flexible et efficient de Base de Données à Base Ontologique pour le Web sémantique et les données techniques*. Université de Poitiers : Thèse de Doctorat.

Franconi, E., F. Baader, U. Sattler, et P. Vassiliadis (200O). *Fundamentals of Data Warehousing, chapter Multidimensional Data Models and Aggregation*. Berlin Heidelberg: Springer Verlag.

Gali, A., C. Chen, K. Claypool, et R. Uceda-Sosa (2004). From ontology to relational databases. *ER Workshops*, 278–289.

Gam, I. et C. Salinesi (2006). A requirement-driven approach for designing data warehouses. *REFSQ'2006*.

Giorgini, P., E. Nicchiarelli, J. Mylopoulos, et R.Sebastiani (2003). Formal reasoning techniques for goal models. *Journal of Data Semantics Springer*.

Giorgini, P., S. Rizzi, et M. Garzetti (2005). Goal-oriented requirement analysis for data warehouse design. *DOLAP'05*.

Glinz, M. (2007). On non-functional requirements. *15th IEEE International Requirements Engineering Conference RE07*, 21–26.

Golfarelli, M. et S. Rizzi (1998). Methodological framework for data warehouse design. *1st ACM International Workshop on Data Warehousing and OLAP*, 3–9.

Golfarelli, M. et S. Rizzi (2009). *Data Warehouse Design: Modern Principles and Methodologies*. McGraw Hill.

Grabova, O., J. Darmont, H. Chauchat, et I. Zolotaryova (2010). Business intelligence for small and middle-sized entreprises. *SIGMOD Record 39*.

Jensen, M. R., T. Holmgren, et T. Pedersen (2004). Discovering multidimensional structure in relational data. *6th International Conference on Data Warehousing and Knowledge Discovery, LNCS Springer 3181*, 138–148.

Khouri, S. et L. Bellatreche (2010a). A methodology and tool for conceptual designing a data warehouse from ontology-based sources. *ACM Thirteenth International Workshop On Data Warehousing and OLAP DOLAP10*.

Khouri, S. et L. Bellatreche (2010b). Ontostar : Outil de conception multidimensionnelle d'un entrepôt de données à base ontologique. *Base de données avancées (26èmes journées) BDA2010*.

Kumar, M., A. Gosain, et Y. Singh (2010). Stakeholders driven requirements engineering approach for data warehouse development. *Journal of Information Processing Systems 6*.

Leavitt, N. (2010). Will nosql databases live up to their promise? *IEEE Computer*, 12–14.

List, B., J. Schiefer, et A. M. Tjoa (2000). Process-oriented requirement analysis supporting the data warehouse design process a use case driven approach. *11th International Conference on Database and Expert Systems Applications DEXA*, 593–603.

Luján-Mora, S. et J. Trujillo (2003). A comprehensive method for data warehouse design. *DMDW03*.

M., B. et A. U. vom Ende (1999). Deriving the initial data warehouse structures from the conceptual data models of the underlying operational information systems. *DOLAP'99*.

Mazon, J., J. Pardillo, E. Soler, O. Glorio, et J. Trujillo (2008). Applying the i* framework to the development of data warehouses. *3rd International i* Workshop istar08*, 79–82.

Moody, D. et M. Kortnik (2000). From enterprise models to dimensionals models: A methodology for data warehouse and data mart design. *International Workshop on Design and Management of Data Warehouses DMDW00*.

Mylopoulos, J., L. Chung, S. Liao, H. Wang, et E. Yu (2001). Exploring alternatives during requirements analysis. *IEEE Software 18*, 92–96.

Narayanan, S., T. Kurc, et J. Saltz (2007). Dbowl: Towards extensional queries on a billion statements using relational databases. *WWW07*.

Nebot, V., R. Berlanga, J. M. Perez, M. J. Aramburu, et T. B. Pedersen (2009). Multidimensional integrated ontologies: A framework for designing semantic data warehouses. *Journal on Data Semantics JoDS 13*, 1–36.

Nuseibeh, B. et S. Easterbrook (2000). Requirements engineering: a roadmap. *Int. Conf. on Soft. Eng. ICSE*, 35–46.

Pan, Z. et J. Heflin (2003). Dldb: Extending relational databases to support semantic web queries. *Workshop on Practical and Scaleable Semantic Web Systems, The 2nd International*

Semantic Web Conference.

Prat, N., J. Akoka, et I. Comyn-Wattiau (2006). A uml-based data warehouse design method. *Decision Support Systems 42*, 1449–1473.

Rizzi, S., A. Abello, J. Lechtenborger, et J. Trujillo (2006). Research in data warehouse modeling and design: dead or alive? *DOLAP*, 3–10.

Rolland, C. (2005). Reasoning with goals to engineer requirements. *Enterprise Information Systems Springer*.

Romero, O., D. Calvanese, A. Abello, et M. Rodriguez-Muro (2009). Discovering functional dependencies for multidimensional design. *12th International Workshop on Data Warehousing and OLAP*, 1–8.

Schiefer, J., B. List, et R. Bruckner (2002). A holistic approach for managing requirements of data warehouse systems. *8th Americas Conference on Information Systems*.

Song, I., R. Khare, et B. Dai (2007). Samstar: A semi-automated lexical method for generating star schemas from an er diagram. *DOLAP07*, 9–16.

Stefanov, V. et B. List (2006). Business metadata for the data warehouse - weaving enterprise goals and multidimensional models. *International Workshop on Models for Enterprise Computing at EDOC*.

Tapucu, D. (2010). *A Generic Model for Handling Preferences in Ontology Based Databases*. Université de Poitiers : Thèse de Doctorat.

Vysniauskas, E. et L. Nemuraite (2006). Transforming ontology representation from owl to relational database. *Information technology and control 35*.

Winter, R. et B. Strauch (2003). A method for demand driven information requirements analysis in data warehousing projects. *36th Hawaii International Conference on Systems Sciences*.

Summary

The current storage structure of a data warehouses (DW) supports the logical model of the warehouse and stores the data according to this model. The importance of a conceptual modelling phase is currently recognized by several studies. The conceptual model provides an abstraction view of the domain and its availability facilitates the interrogation of the physical model by end users. The conceptual and logical models of the warehouse are defined from two components: data sources and users requirements. The involvement of user's requirements is recognized by practitioners and by the research community as an essential first step determining the success of the project. Requirements are besides used at different stages of exploitation of the DW (optimization, personalization, recommendations, traceability, etc.). We note, however, that no trace of the conceptual model or the requirements model is stored in the final warehouse. We propose in this paper a new storage structure of DWs representing these three models: the conceptual model, the requirements model and the logical model, persistently in the warehouse.