Enrichissement de l'architecture ANSI/SPARC pour expliciter la sémantique des données: une approche fondée sur les ontologies.

Chimène Fankam*, Stéphane Jean*, Guy Pierra*, Ladjel Bellatreche*

*LISI / ENSMA - Université de Poitiers
86961 Futuroscope Cedex
{fankamc,jean,pierra,bellatreche}@ensma.fr,

Résumé. L'architecture de bases de données ANSI/SPARC a été principalement définie pour permettre l'accès aux données indépendamment de leur représentation physique. La conception d'une base de données selon cette architecture passe par la transformation d'un modèle conceptuel en un modèle logique. Cette transformation peut engendrer une perte de sémantique de données. Ce qui pose des problèmes d'intégration et d'échange de plusieurs bases de données, ou de génération des interfaces d'accès aux données pour un utilisateur final. En tant que modèle permettant d'exprimer la sémantique des données, les ontologies constituent une solution pertinente à ces problèmes. Dans cet article, nous montrons le besoin d'étendre l'architecture ANSI/SPARC par l'ajout d'un niveau ontologique, permettant de conserver les ontologies décrivant la sémantique des données contenues dans une base de données. Notons que cette extension n'aura aucune incidence sur les applications conçues autour de l'architecture initiale. Nous analysons cette architecture en termes de besoins d'exploitation puis discutons de l'implantation d'une telle architecture.

1 Introduction

Actuellement les bases de données représentent l'outil principal de stockage de données permettant aux utilisateurs d'accéder simultanément aux données. Construire une base de données passe souvent par plusieurs étapes : (1) la modélisation conceptuelle, (2) la modélisation logique, (3) la modélisation physique et (4) la modélisation externe. Un modèle conceptuel est une représentation de la connaissance propre à un domaine. Il est caractérisé par (a) le domaine auquel il s'intéresse, (b) le formalisme qui a permis de le définir (modèle de Chen - connu sous le nom de modèle Entité-Association), et enfin (c) le contexte ou le point de vue qu'il souhaite représenter, et qui définit les questions auxquelles il vise à répondre. Le modèle conceptuel est traduit vers un modèle logique qui est une spécification des données telle quelle sera implémentée sur le système de base de données. Cette traduction passe par l'application de certaines règles. Le modèle physique définit la façon selon laquelle sont stockées les données et les méthodes pour y accéder (tous les mécanismes d'indexation par exemple). Des modèles externes ou les vues utilisateurs permettent d'adapter les données fournies aux

besoins des différentes catégories d'utilisateurs. Cette démarche de conception de bases de données suit parfaitement l'architecture ANSI/SPARC proposée par Charles Bachman Bachman (1974). La mise en oeuvre du processus de modélisation des bases de données suivant l'architecture ANSI/SPARC présente deux inconvénients majeurs Dehainsala et al. (2007b) :

1. la très forte dépendance des modèles vis-à-vis des concepteurs et des besoins applicatifs particuliers. En effet, deux modèles conceptuels conçus par deux concepteurs différents visant à remplir les même fonctions seront soit (1) partiellement différents du point de vue du domaine exact modélisé, soit (2) très différent du point de vue de la structure du modèle résultant. Les modèles logiques générés seront d'autant plus différents (conflits de granularité des informations, le nommage des concepts, les types des données, etc.).

2. l'écart existant en général entre les modèles conceptuels et les modèles logiques des données, qui s'accroît avec la divergence des formalismes. En effet, le passage du modèle conceptuel au modèle logique nécessite des opérations de traduction complexes. Le modèle logique résultant de cette traduction est alors très différent du modèle conceptuel initial (les entités et les associations sont éclatées en de multiples tables dans le cas des bases de données relationnelles). Le modèle logique résultant peut, très vite devenir incompréhensible (surtout pour des grands modèles conceptuels) pour un utilisateur et ne plus permettre d'appréhender le problème initial traité.

Au centre de ces problèmes se pose la question de la pérennisation du modèle conceptuel qui, en plus se doit d'être consensuel afin de pouvoir résoudre les problèmes d'intégration (différents conflits sémantiques et schématiques) Bellatreche et al. (2004). Il est donc apparu intéressant de donner plus de place et d'autonomie à la notion de modèle conceptuel pour permettre sa représentation effective dans une base de données Sugumaran et Storey (2006), Hondjack (2007). C'est ce que va permettre la modélisation à base ontologique qui introduit un niveau supplémentaire : le niveau ontologie. Dans cette nouvelle approche, le développement d'un modèle conceptuel (MC) est précédé par le développement d'une ontologie utilisant un formalisme standard de modélisation d'ontologie permettant de prendre en compte la sémantique des concepts dans le processus de modélisation des bases de données. Dans ce cas, le MC est alors vu comme un sous-ensemble de cette ontologie et, ontologie et données sont toutes deux représentées dans la base de données. Nous appelons une telle base de données, *une base de données à base ontologique*. D'où notre proposition d'étendre l'architecture ANSI/SPARC afin qu'elle puisse supporter les bases de données à base ontologique.

Cet article comprend 6 sections. La section 2 présente une comparaison entre les ontologies et les modèles conceptuels et une classification des ontologies de domaine, en proposant un modèle d'oignon. La section 3 propose notre extension de l'architecture ANSI/SPARC par le niveau ontologique. La section 4 présente les exigences résultant de la décomposition du niveau ontologique de notre architecture selon les trois couches du modèle en oignon. La section 5 s'étale sur l'un des problèmes de bases de données à base ontologique, qui est, la représentation des concepts non canoniques et propose certaines solutions. Enfin, la section 6 donne une conclusion, récapitule les principaux résultats et suggère quelques perspectives.

Enrichissement de l'architecture ANSI/SPARC pour expliciter la sémantique des données.

2 Ontologies et bases de données

Dans cet article, nous nous basons sur l'interprétation de la notion d'ontologie que nous avons présenté dans Jean et al. (2007). Nous rappelons brièvement cette interprétation dans cette section.

Dans nos travaux, une ontologie de domaine est un *dictionnaire formel et consensuel des catégories et propriétés d'entités existant dans un domaine d'étude et des relations qui les lient*. Cette définition met en avant trois caractéristiques qui distinguent une ontologie de domaine des autres modèles informatiques. Une ontologie est (1) *formelle* permettant ainsi des raisonnements (automatiques ou non) et de la vérification de consistance (2) *consensuelle* dans une communauté et (3) *référençable*, c'est-à-dire que toute entité ou relation décrite dans l'ontologie peut être directement référencée par un symbole, dans n'importe quel but et à partir de n'importe quel contexte, indépendamment des autres entités et relations.

2.1 Comparaison entre ontologie de domaine et modèle conceptuel

Les ontologies et les modèles conceptuels présentent à la fois des similitudes et des différences Spyns et al. (2002); Hondjack (2007).

2.1.1 Similitudes

Comme les modèles conceptuels (MC), les ontologies conceptualisent également l'univers du discours au moyen de classes associées à des propriétés et hiérarchisées par subsomption. Les principes de bases de la modélisation sont similaires.

2.1.2 Différences

On peut identifier cinq différences majeures entre ces deux types de modèles.

1. *L'objectif de la modélisation*. Les MCs prescrivent l'information qu'ils représentent dans un système informatique particulier. Au contraire, les ontologies décrivent les concepts d'un domaine indépendamment de toutes applications et systèmes informatiques particuliers dans lesquels l'ontologie pourrait être utilisée.
2. *L'identification des concepts*. Les classes et les propriétés définies dans les ontologies sont associées à des identifiants, ce qui leur permet d'être référencées à partir de n'importe quel format ou modèle indépendamment de leur structure. Au contraire, la conceptualisation effectuée dans un MC ne peut pas être réutilisée à l'extérieur et indépendamment de ce MC.
3. *Le raisonnement*. Le caractère formel des ontologies permet d'appliquer des opérations de raisonnement sur les ontologies soit pour vérifier la cohérence des informations, soit pour déduire de l'information Fra (2003). Par exemple dans la plupart des modèles d'ontologies Antoniou et van Harmelen (2003), Pierra (2003), Kifer et al. (1995), pour une ontologie et une classe données, on peut calculer (1) toutes ses super-classes (directes ou non), (2) ses propriétés caractéristiques (héritées ou locales), (3) toutes ses instances (polymorphes ou locales), etc.

4. *La consensualité*. Le caractère consensuel des ontologies permet de représenter de la même façon les mêmes concepts dans tous les systèmes d'une "communauté".
5. *La souplesse de description*. Toutes les instances des classes d'une ontologie, peuvent ne pas initialiser les mêmes propriétés. Elles n'ont pas forcement la même structure. Cette souplesse dans la description des instances est permise par le fait que les concepts des ontologies soient associés à des identifiants universels. Cela a pour conséquence de rendre les ontologies beaucoup plus simples à utiliser pour des échanges ou intégration de systèmes informatiques.

2.2 Classification des ontologies de domaine : le modèle en oignon

Toutes les ontologies de domaine ne sont pas identiques. Nous distinguons les trois catégories d'ontologies suivantes :

- les *Ontologies Conceptuelles Canoniques* (OCC) contenant les ontologies dont les définitions ne contiennent aucune redondance. Les OCC (Dublincore, IEC-61360-44 (International Electronic Commision)) adoptent une approche de structuration de l'information en termes de classes et de propriétés et leur associent des identifiants réutilisable dans différents langages ;
- les *Ontologies Conceptuelles Non Canoniques* (OCNC) contenant les ontologies contenant non seulement des concepts primitifs (canoniques), mais aussi des concepts définis (non canoniques), c'est-à-dire qui peuvent être construits à partir de concept primitifs et/ou définis à l'exemple de OWL ;
- les *Ontologies Linguistiques* (OL) contenant les ontologies qui définissent l'ensemble des termes qui apparaissent dans la description langagière d'un domaine. Outre des définitions textuelles, un certain nombre de relations linguistiques (synonyme, hyperonyme, hyponyme, etc.) sont utilisées pour capturer de façon approximative et semi formelle les relations entre les mots. Un exemple d'OL est WordNet.

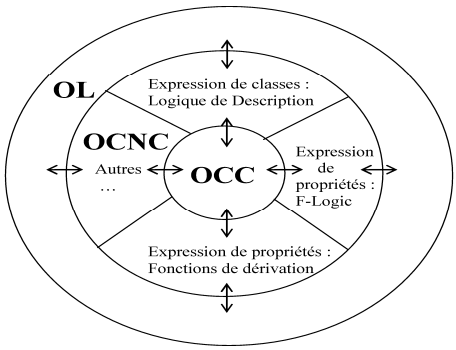

FIG. 1 – *Le Modèle en oignon*

Ces trois types d'ontologies peuvent être combinés dans un modèle en couches que nous appelons le *modèle en oignon* (voir Figure 1). A la base de ce modèle se situe une OCC. Elle fournit une base formelle pour modéliser et échanger efficacement la connaissance d'un domaine. A partir des concepts primitifs de l'OCC, une OCNC peut être construite. Cette OCNC fournit

les mécanismes pour lier différentes conceptualisations faites sur ce domaine. Finalement, une représentation en langage naturel des concepts de l'OCNC, éventuellement dans les différents langages où ces concepts sont significatifs, peut être fournie par une OL.

Chaque catégorie d'ontologies offre des capacités particulières :

- les OCC fournissent une description canonique et précise de chaque concept d'un domaine donné. Elles fournissent une base solide pour l'échange entre différentes sources d'information ;
- les opérateurs des OCNC sont utilisés pour interagir avec ou intégrer d'autres applications ou sources ayant déjà leur propre ontologie ;
- les OL offrent des capacités linguistiques sur l'ensemble des concepts (primitifs et définis) du domaine.

Nous montrons dans la section suivante en quoi ces caractéristiques sont intéressantes pour les bases de données.

2.3 Liens entre le modèle en oignon et les bases de données

Chaque couche du modèle en oignon peut être utilisé pour résoudre différents problèmes des bases de données.

- Les OCC sont des MCs formels et partageables. Ils peuvent être utilisés comme base pour la conception d'un modèle logique (ML) de base de données ou comme schéma global dans un scénario d'intégration de bases de données.
- Les OCNC proposent des mécanismes similaires aux vues des bases de données, avec une théorie formelle offrant des capacités d'inférence. Ces mécanismes peuvent être utilisés pour réaliser le mapping entre différents schémas de bases de données, Ou pour adapter le schéma de la base de données aux besoins des différentes catégories d'utilisateurs
- Les OLs peuvent être utilisées pour localiser les similitudes existantes entre plusieurs schémas de bases de données Beneventano et al. (2000), pour documenter les Systèmes de Gestion de Bases de Données (SGBD) existants ou pour enrichir le langage de dialogue personne/SGBD.

L'utilisation croissante d'ontologies dans divers domaines (techniques ou documentaires) fait qu'aujourd'hui, certaines données sont déjà représentées comme des instances de classes d'ontologies et ont donc leur sémantique définie à partir d'ontologies. Nous appellerons ces données des données à base ontologique. Le besoin de représenter les ontologies et les données à base ontologique dans de vraies bases de données plutôt qu'en mémoire centrale ou dans des fichiers ordinaires se fait de plus en plus sentir. Ces besoins sont à l'origine de l'émergence d'une nouvelle approche de bases de données dites bases de données à base ontologique.

2.4 Bases de Données à Base Ontologique (BDBO)

De plus en plus de données (ou de métadonnées) sont décrites par référence à des ontologies. La taille croissante de telles données rend nécessaire de les gérer au sein de bases de données originales, que nous appelons bases de données à base ontologique (BDBO), et qui possèdent la particularité de représenter, outre les données, les ontologies qui en définissent le sens. L'approche de modélisation utilisant les ontologies ajoute un niveau supplémentaire (niveau ontologique) aux trois niveaux de l'architecture traditionnelle de modélisation des bases

de données (niveau conceptuel, niveau logique, niveau externe). Dans l'approche basée sur les ontologies, le MC est un sous-ensemble (ou éventuellement une spécialisation Bellatreche et al. (2004)) de l'ontologie. Une BDBO représentera donc explicitement (1)les ontologies, (2)la structure des données,(3)les données proprement dites, ainsi que,(4)le lien entre les données et leur schéma et celui entre les données et l'ontologie. Ce type de base de données permet l'accès aux données au niveau sémantique, c'est-à-dire au niveau ontologique.
Comme nous l'avons vu dans le modèle en oignon, les OCNC et les OL introduisent des extensions aux OCC. En particulier, la possibilité de définir des concepts dit définies en introduisant des équivalences conceptuelles. les systèmes de gestion des ontologies s'appuyant sur de telles ontologies (SESAME, ONTOMS, ...) présentent des inconsistances du point de vue des bases de données classiques :

- la duplication des données qui appartiennent à plusieurs classes distinctes de la fois. En effet les instances des classes définies sont aussi des instances de classes primitives et donc il y a duplication de l'information. La même information se trouve à la fois dans l'extension de la classe primitive et dans celles des classes définies à partie d'elle.
- fiabilité des données. De nombreuses opérations sont nécessaires pour assurer la consistance des données suite à une mise à jour. C'est le cas par exemple lors de la suppression d'une instance d'un concept primitif ; on aurait alors besoin de connaître les classes définies qui la référence afin de mettre à jour leur extension.

Ces exemples montrent la difficulté d'avoir une véritable base de données (non déductive) recouvrant intégralement tous types d'ontologie. Afin d'aboutir à de véritables systèmes de gestion pour les données à base ontologique à l'image des SGBDs classiques, il est important de séparer le canonique du non canonique dans les ontologies afin de :

- définir un système canonique et non redondant de gestion pour les concepts canoniques des ontologies et de leurs instances.
- définir des mécanismes d'extension de ce système permettant la représentation des concepts non canoniques et des autres extensions propres aux OCNC et aux OL tout en préservant l'intégrité et la non redondance des données dans le système.

L'architecture ANSI/SPARC ne permet pas de supporter directement les BDBO qui introduisent un niveau supplémentaire d'accès aux données : le niveau ontologique permettant d'intégrer dans la base de données à la fois les données mais également leur sémantique. Ces observations nous ont conduit à proposer une extension de cette architecture que nous présentons dans la section suivante.

3 Application des ontologies aux bases de données : proposition de l'extension de l'architecture ANSI/SPARC

Un des buts essentiels d'une base de données est d'une part d'assurer une gestion efficace des données et d'autre part de permettre l'accès aux données indépendamment de leur représentation physique. L'architecture ANSI/SPARC ANSI/X3/SPARC (1975) a été proposée pour remplir ces objectifs. Elle distingue les deux niveaux d'accès suivants :

- *le niveau physique*. Il définit comment les données sont stockées et gérées en utilisant le système de gestion de fichiers ;

– *le niveau logique*. Il définit comment les données sont structurées en utilisant le modèle de données de la base de données (par exemple, le modèle relationnel ou objet).

La conception d'une base de données suivant cette architecture s'accompagne d'une perte de sémantique des données liée à la transformation du MC en ML.

En tant que modèle permettant d'exprimer la sémantique des données, les ontologies sont une solution pertinente à ces problèmes. Nous proposons en conséquence l'extension de l'architecture ANSI/SPARC avec le *niveau ontologique*. Ce niveau définit la sémantique des données. Il est constitué des descriptions sémantiques fournies par une ontologie. Cette architecture étendue est présentée sur la figure 2.

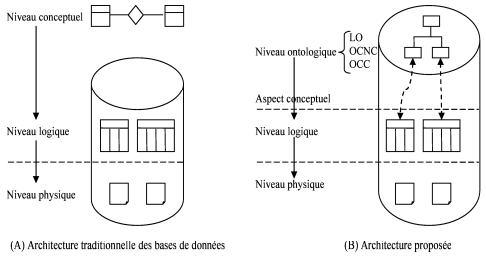

FIG. 2 – *Notre proposition d'architecture de base de données*

L'architecture traditionnelle de base de données est présentée dans la partie (A) de cette figure, située à gauche. Dans cette approche, un MC, représenté dans un formalisme tel que le modèle Entité/Relation est conçu. Il est ensuite souvent utilisé pour générer automatiquement le ML des données constitué d'un ensemble de tables dans les SGBD relationnels ou relationnels-objets. Ce modèle logique est lui même représenté au niveau physique à l'aide d'un ensemble de fichiers.

Dans la partie (B), nous proposons l'extension de cette architecture en intégrant les deux éléments suivants :

– *le niveau ontologique*. Il est composé d'ontologies qui définissent les concepts de différents domaines d'étude sous la forme de classes et de propriétés, indépendamment des besoins applicatifs de la base de données. Cependant, ces ontologies peuvent être spécialisées pour représenter les éventuels concepts manquants par rapport à ces besoins. Lorsqu'elles sont conçues selon le modèle en oignon, ces ontologies comportent toujours une couche canonique. Elles peuvent éventuellement comporter une couche non canonique. Elles comportent toujours un minimum d'aspects linguistiques et, en particulier, des termes qui dénotent les concepts représentés ;
– *l'aspect conceptuel*. Cet aspect est représenté par le lien entre le niveau ontologique et le niveau logique. Ce lien indique le sous-ensemble de concepts (classes et propriétés) des ontologies qui sont exploités pour satisfaire les besoins des applications pour lesquelles la base de données est conçue. Cet ensemble de concepts, une fois choisi, peut être utilisé pour générer automatiquement le modèle logique des données.

Les trois niveaux de cette architecture (physique, logique et ontologique) sont actuellement implantés dans différentes base de données qui permettent de stocker à la fois des ontologies et des données Broekstra et al. (2002); Alexaki et al. (2001); B.McBride (2001); Dehainsala

et al. (2007a); Park et al. (2007); Pan et Heflin (2003); L.Ma et al. (2004); Bozsak et al. (2002); Harris et Gibbins (2003); Stoffel et al. (1997). Nous appelons Bases de Données à Base Ontologique (BDBO), les bases de données présentant cette caractéristique. Cependant, peu de travaux se sont intéressés à tirer profit des différents niveaux de cette architecture. Nous montrons ceci dans la section suivante en identifiant les besoins d'exploitation induits par l'architecture de bases de données proposée.

4 Exigences induites par les besoins d'exploitation de l'architecture de bases de données proposée

4.1 Exigences liées au modèle en oignon

La première grande innovation de l'architecture ANSI/SPARC traditionnelle est la distinction entre la représentation interne des données au niveau physique et la représentation logique de celles-ci. Cette distinction permet de définir, manipuler et interroger les données au niveau logique indépendamment de leur implantation physique.

L'extension de cette architecture que nous proposons permet également la distinction entre la représentation logique des données (structure) et la représentation ontologique de celles-ci (sémantique). Cette distinction permet de définir, manipuler et interroger les données au niveau ontologique indépendamment de leur représentation logique. Cette capacité permet ainsi d'interroger différentes BDBO, utilisant la même ontologie mais des schémas différents, avec les mêmes requêtes (*requêtes ontologiques*).

Exigence 1 (*Traitement des données au niveau ontologique*)

L'environnement doit permettre d'exprimer des requêtes sur les données, indépendamment de leur schéma, à partir des ontologies contenues dans une BDBO.

La seconde grande innovation de l'architecture ANSI/SPARC est la possibilité de créer des schémas externes (vues) dans une base de données. L'architecture ANSI/SPARC permet ainsi la définition de plusieurs schémas externes représentant différentes vues sur la base de données avec des possibilités de recouvrement. Des requêtes peuvent être exprimées sur ces schémas externes, le SGBD se chargeant d'interpréter ces requêtes en fonction du schéma logique des données.

Dans l'extension que nous proposons, la couche OCNC d'une ontologie offre les mêmes capacités que les vues au niveau ontologique. Elle permet à chaque utilisateur de définir sa propre perception du domaine d'étude en représentant un ensemble de concepts non canoniques à partir des concepts canoniques de l'ontologie. La définition de tels concepts ainsi que la possibilité de les utiliser dans les requêtes est donc une exigence d'un langage d'exploitation pour cette architecture.

Enrichissement de l'architecture ANSI/SPARC pour expliciter la sémantique des données.

Exigence 2 (*Définition de concepts non canoniques*)

> *L'environnement doit permettre de définir des concepts non canoniques à partir des concepts canoniques d'une ontologie. Il doit également permettre d'exprimer des requêtes ontologiques à partir de ces concepts non canoniques, se chargeant ainsi d'interpréter ces requêtes en fonction des concepts canoniques associés.*

La dernière couche du modèle en oignon est constituée de la partie LO. Lorsqu'une ontologie est construite selon le modèle en oignon, sa partie LO associe à chacun de ses concepts un nom sous forme de terme, et une définition textuelle. La définition permet notamment à des êtres humains de comprendre l'ontologie et les noms permettent d'y faire référence. Or, le contexte d'utilisation d'une ontologie est souvent international. Par exemple, un des objectifs du Web Sémantique est de favoriser l'échange d'informations contenues sur le Web qui est un outil international. Donc, une exigence essentielle est que le système puisse supporter des LO multilingues et que le langage permette de référencer les éléments de l'ontologie directement par leurs dénominations dans leurs propres langues.

Exigence 3 (*Exploitation linguistique*)

> *L'environnement doit permettre d'exploiter les dénominations linguistiques, éventuellement données dans plusieurs langues naturelles, qui peuvent être associées à chaque concept d'une ontologie.*

Nous venons de présenter les exigences résultant de la décomposition du niveau ontologique de notre architecture selon les trois couches du modèle en oignon. Notre architecture présente également la particularité de vouloir être compatible avec l'architecture traditionnelle des bases de données. Ceci induit de nouvelles exigences.

4.2 Exigences liées à la compatibilité avec l'architecture traditionnelle des bases de données

Notre architecture étant basée sur l'architecture ANSI/SPARC, elle intègre le niveau logique. Le langage doit donc non seulement permettre d'accéder aux données à partir du niveau ontologique (section précédente) mais également au niveau logique. Or, le langage SQL a été défini pour manipuler les données à ce niveau. Afin de bénéficier de la vaste adoption de ce langage, ainsi que des nombreux travaux menés sur l'optimisation de requêtes SQL dans les SGBD, le langage doit être compatible avec le langage SQL.

Exigence 4 (*Compatibilité avec SQL*)

> *L'environnement doit permettre d'accéder au niveau logique d'une BDBO en étant compatible avec le langage SQL.*

L'inconvénient de construire une architecture à plusieurs niveaux telle que celle que nous proposons est de complexifier les traitements sur les données au fur et à mesure que l'on monte dans les niveaux. La complexité introduite a généralement un impact sur l'efficacité de ces traitements. Dans de telles architectures, une manière d'optimiser les traitements à un niveau donné, est d'accéder au niveau inférieur. Par exemple, dans l'architecture ANSI/SPARC, pour

optimiser certaines requêtes SQL portant sur le niveau logique de cette architecture, on peut utiliser la connaissance que l'on a du niveau physique avec l'utilisation des HINTS proposés par la plupart des SGBDs commerciaux. Un hint est une directive pour forcer l'optimiseur de requête de choisir un traitement spécifique d'une requête donnée afin de l'optimiser. Par exemple, ORACLE permet de forcer l'optimiseur de requêtes à utiliser un index donné (syntaxe SELECT `/*+ INDEX (nom_table nom_index) */ FROM ....`). ORACLE permet également d'accéder au niveau physique d'une base de données dans le langage de définition de données proposé. Ainsi, une table peut être créée en indiquant le *tablespace* dans lequel les données qui correspondent à cette table seront stockées.

Dans l'architecture que nous proposons nous avons ajouté le niveau ontologique au dessus du niveau logique. Donc, afin de permettre l'optimisation des traitements à ce niveau, le langage doit permettre d'accéder au niveau inférieur, c'est-à-dire le niveau logique.

Exigence 5 (*Définition, manipulation et interrogation du schéma des données*)

L'environnement doit permettre de définir, manipuler et rechercher le schéma des données à partir de l'ontologie.

La figure 3 montre comment les exigences que nous avons définies se positionnent par rapport à l'architecture que nous proposons. Les exigences 1, 2 et 3 requièrent de pouvoir effectuer des requêtes ontologiques construites à partir des définitions canoniques, non canoniques et linguistiques que peut fournir une ontologie. Les exigences 4 et 5 concernent le niveau logique de notre architecture. Elles requièrent d'une part de pouvoir utiliser le langage SQL pour manipuler ce niveau et d'autre part de pouvoir passer du niveau ontologique au niveau logique afin de permettre l'optimisation des traitements.

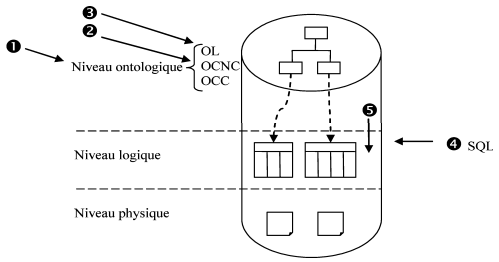

FIG. 3 – *Positionnement des exigences par rapport à l'architecture de base de données proposée*

Les exigences définies dans cette section constituent une spécification de besoins concernant un langage d'exploitation de l'architecture ANSI/SPARC que nous avons proposée et les outils qui doivent lui être associés afin d'en faciliter l'utilisation. Un langage répondant à cet ensemble d'exigences, permettrait d'exploiter pleinement cette architecture.

5 La représentation des concepts non canoniques

Le principal problème posé par la définition des concepts non canoniques est que les constructeurs fournis par les modèles d'ontologies sont divers. Les constructeurs des modèles d'ontologies considérés permettant de définir des concepts non canoniques sont présentés dans le tableau 1. Ces constructeurs peuvent être regroupés en trois catégories selon leur origine :

- les constructeurs de classes et de propriétés définies. Ces constructeurs sont issus de la logique de description. Les expressions logiques liées à ces constructeurs (par exemple, les expressions booléennes) peuvent être exploitées par un moteur d'inférence, aussi appelé raisonneur (par exemple, Racer Haarslev et Möller (2001)), afin, notamment, d'organiser les concepts canoniques et non canoniques dans une seule et même hiérarchie ;
- les règles logiques. Ces constructeurs sont issus de la logique des frames. Ils nécessitent l'utilisation d'un moteur de règles (par exemple, Jess[1] pour déduire de nouveaux faits à partir des faits connus ;
- les expressions algébriques (exemple, diamètre = rayon * 2). Ces constructeurs sont issus de la communauté du traitement des données. Ils nécessitent un interpréteur d'expressions algébriques.

Constructeur	**Modèle d'ontologies**
Construction de classes par des expressions booléennes (Union, Intersection, Complément)	OWL
Construction de classes comme des restrictions (de domaine, de valeur ou de cardinalité)	OWL
Caractéristiques des propriétés (inverse, symétrique, transitive)	OWL
Règles logiques	F-Logic
Expression algébrique	PLIB

TAB. 1 – *Constructeurs non canoniques proposés par les modèles d'ontologies*

Cette diversité d'approches soulève une question : quel(s) type(s) de constructeurs doivent être utilisés pour définir les concepts non canoniques d'une ontologie ? Cette question a soulevé un débat entre les différentes communautés Motik et al. (2006); de Bruijn et al. (2005); Patel-Schneider et Horrocks (2006). De notre point de vue, ce débat a montré que les différents constructeurs proposés ont chacun leur utilité. Ils sont plus ou moins adaptés selon le domaine d'application dans lesquels ils sont utilisés. L'objectif est donc maintenant de proposer une architecture de BDBO suffisamment flexible pour permettre d'utiliser ces différents constructeurs.

Trois alternatives principales peuvent être envisagées pour la prise en compte des concepts non canoniques dans les bases de données.

1. La modélisation par les bases de données déductives où de nombreux mécanismes basées sur des règles déductives sont intégrés et, permettent de dériver de nouvelles infor-

[1] `http://herzberg.ca.sandia.gov/jess/`

mations à partir de l'information existante, ou de tester la cohérence de l'information contenue dans la base de données.

2. La connexion d'une base de données classique à un raisonneur externe qui se chargera à posteriori de dériver de nouvelles informations à la suite des requêtes émises sur la base de données. La mise en oeuvre de cette solution est complexe et nécessite l'adoption d'une politique efficace et complète d'extraction des données de la base de données afin de fournir au raisonneur toute l'information dont il a besoin pour chaque requête donnée. Aussi, dans le contexte actuel des applications en réseaux, il se pose le problème de la gestion du flux de données qui va transiter sur le réseau ainsi que l'éventuel surcharge de celui-ci.
3. L'adoption d'une gestion à priori des concepts définis qui suppose que la base de données est toujours complète. La charge est ici laissée aux applications de toujours fournir des informations complètes à la base de données. Cette dernière solution que nous avons adoptée permet une prise en compte plus aisée des concepts non canoniques par l'exploitation des caractéristiques de l'architecture ANSI/SPARC.

Du point de vue des bases de données, les ontologies non canoniques introduisent la redondance nécessitant des traitements particuliers. Cela accroît la difficulté d'avoir une véritable base de données (non déductive) recouvrant le modèle en oignon. Afin d'aboutir à de véritables systèmes de gestion des données à base ontologique, il est important d'avoir une gestion particulière à chaque niveau du modèle en oignon.

5.1 Classes définies

La principale difficulté ici est de ne pas introduire la redondance au niveau des données ou des instances de classes. Les classes définies étant construites en utilisant les opérateurs ensemblistes (union, intersection) et, les opérateurs de restrictions ; il est envisageable de pouvoir dans la BDBO, ne pas représenter leur extensions, mais plutôt, la calculer. Ceci est possible en exploitant comme dans les bases de données classiques le mécanisme de vue qui permet d'effectuer des unions, intersection, sélection sur des ensembles de données.

5.2 Caractéristiques algébriques des relations

- Symétrique : nous proposons de saturer automatiquement et à posteriori les données. Ce mécanisme est simple à mettre en oeuvre et ne pose pas de problème lors de la mise à jour des données. La seule restriction concerne la suppression qui doit être contrôlée. En effet il ne doit pas être permis de supprimer ou de mettre à jour la valeur d'une propriété symétrique car la base de données deviendrait ainsi inconsistante.
- Transitive : c'est une caractéristique complexe à mettre en oeuvre dans une base de données. En effet, celle-ci pose plusieurs problèmes. Le mécanisme de saturation à l'insertion ou à posteriori, induit un coût lié au temps de calcul de la fermeture transitive. Si l'on considère par exemple être en présence dans la base de données d'un couple $P(x, y)$, alors l'insertion d'un nouveau couple $P(y, z)$ entraînera automatiquement l'insertion du couple $P(x, z)$. Cependant, à la suppression du couple $P(y, z)$ on ne peut faire aucune hypothèse sur l'origine de couple $P(x, z)$ et donc aucune règle ne permet de la supprimer. Ainsi, la saturation pour la caractéristique transitive est un mécanisme

complexe à mettre en oeuvre. La solution que nous proposons d'adopter est de se baser sur une saturation à priori. Nous supposons au départ que les données fournit sont saturées ; ainsi à l'ajout d'une nouvelle relation portant sur une propriété transitive, la BDBO va être saturée par un trigger de manière non récursive. Pour cela, à l'ajout d'une nouvelle relation P(x,y), il suffit pour toute relation existante P(i,x) dans la base de données, d'introduire une nouvelle relation P(i,y).

5.3 Expressions algébriques

Comme pour les classes définies, la valeur d'une propriétés dérivée est calculée à partir l'évaluation de son expression. Cette évaluation pourra être encapsulée par une vue.

6 Conclusion et Perspectives

Nous avons présenté dans cet article la notion d'ontologie et son intégration dans une architecture de bases de données appelées bases de données à base d'ontologies (BDBO). Les BDBO apportent un niveau supplémentaire à la modélisation classique des bases de données en ajoutant le niveau ontologique qui représente la sémantique des données et la possibilité d'accéder aux données à partir de ce niveau supplémentaire. Les BDBOs apportent également des solutions aux problèmes de la modélisation classique en dérivant le modèle conceptuel comme un sous-ensemble de l'ontologie. Cette possibilité n'étant pas disponible dans l'architecture classique de bases de données ANSI/SPARC, nous avons proposé son extension afin de supporter les BDBOs. Cette extension ne modifie pas cette architecture et les applications conçues autour d'elle continuent à exister préservant ainsi le principe de compatibilité ascendante. Nous avons discuté des exigences de cette extension par rapport au modèle en oignon de classification des ontologies et à la compatibilité avec l'architecture traditionnelle des bases de données.

Comme perspectives à ce travail, nous envisageons de poursuivre la validation cette architecture. Cette validation concerne deux aspects importants : (1) assurer le bon fonctionnement des applications conçues autour de cette architecture ainsi que celles conçues autour de l'architecture initiale (compatibilité ascendante), et, (2) garantir une meilleure performance.

Références

(2003). The description logic handbook : Theory, implementation, and applications. In F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, et P. F. Patel-Schneider (Eds.), *Description Logic Handbook*. Cambridge University Press.

Alexaki, S., V. Christophides, G. Karvounarakis, D. Plexousakis, et K. Tolle (2001). The ICS-FORTH RDFSuite : Managing voluminous RDF description bases. In *Proceedings of the 2nd International Workshop on the Semantic Web*.

ANSI/X3/SPARC (1975). Study group on data management systems, interim report. *Bulletin of ACM SIGMOD 7*(2).

Antoniou, G. et F. van Harmelen (2003). Web ontology language : Owl.

Bachman, C. W. (1974). Summary of current work - ansi/x3/sparc/study group - database systems. Volume 6, pp. 16–39.

Bellatreche, L., G. Pierra, D. Nguyen Xuan, H. Dehainsala, et Y. Ait Ameur (2004). An a priori approach for automatic integration of heterogeneous and autonomous databases. *International Conference on Database and Expert Systems Applications (DEXA'04)*, 475–485.

Beneventano, D., S. Bergamaschi, S. Castano, A. Corni, R. Guidetti, G. Malvezzi, M. Melchiori, et M. Vincini (2000). Information integration : The momis project demonstration. In *Proceedings of 26th International Conference on Very Large Data Bases (VLDB'00)*, pp. 611–614. Morgan Kaufmann.

B.McBride (2001). Jena : Implementing the rdf model and syntax specification. *Proceedings of the 2nd International Workshop on the Semantic Web*.

Bozsak, E., M. Ehrig, S. Handschuh, A. Hotho, A. Maedche, B. Motik, D. Oberle, C. Schmitz, S. Staab, L. Stojanovic, N. Stojanovic, R. Studer, G. Stumme, Y. Sure, J. Tane, R. Volz, et V. Zacharias (2002). Kaon - towards a large scale semantic web. In *Proceedings of the 3rd International Conference on E-Commerce and Web Technologies (EC-WEB'02)*, London, UK, pp. 304–313. Springer-Verlag.

Broekstra, J., A. Kampman, et F. van Harmelen (2002). Sesame : A generic architecture for storing and querying rdf and rdf schema. In I. Horrocks et J. Hendler (Eds.), *Proceedings of the 1st International Semantic Web Conference (ISWC'02)*, Number 2342 in Lecture Notes in Computer Science, pp. 54–68. Springer Verlag.

de Bruijn, J., R. Lara, A. Polleres, et D. Fensel (2005). Owl dl vs. owl flight : conceptual modeling and reasoning for the semantic web. In A. Ellis et T. Hagino (Eds.), *Proceedings of the 14th international conference on World Wide Web (WWW'05)*, pp. 623–632. ACM.

Dehainsala, H., G. Pierra, et L. Bellatreche (2007a). Ontodb : An ontology-based database for data intensive applications. In *Proceedings of the 12th International Conference on Database Systems for Advanced Applications (DASFAA'07)*, Lecture Notes in Computer Science, pp. 497–508. Springer.

Dehainsala, H., G. Pierra, L. Bellatreche, et Y. Aït Ameur (2007b). Conception de bases de données à partir d'ontologies de domaine : Application aux bases de données du domaine technique. In *Proceedings of 1ères Journées Francophones sur les Ontologies (JFO'07)*, pp. 215–230.

Haarslev, V. et R. Möller (2001). Description of the racer system and its applications. In *Working Notes of the 2001 International Description Logics Workshop (DL'01)*.

Harris, S. et N. Gibbins (2003). 3store : Efficient bulk rdf storage. In *Proceedings of the 1st International Workshop on Practical and Scalable Semantic Systems (PPP'03)*.

Hondjack, D. (2007). Explicitation de la sémantique dans les bases de donnéesă : base de données à base ontologique et le modèle ontodb. *Thèse de Doctorat Université de Poitiers*.

Jean, S., G. Pierra, et Y. Aït-Ameur (2007). *Domain Ontologies : a Database-Oriented Analysis*, Volume 1 of *Lecture Notes in Business Information Processing*, pp. 238–254. Springer Berlin Heidelberg.

Kifer, M., G. Lausen, et J. Wu (1995). Logical foundations of object-oriented and frame-based languages. Volume 42, pp. 741–843.

L.Ma, Z. Su, Y. Pan, L. Zhang, et T. Liu (2004). Rstar : an rdf storage and query system for enterprise resource management. pp. 484 – 491.

Motik, B., I. Horrocks, R. Rosati, et U. Sattler (2006). Can OWL and logic programming live together happily ever after ? In *Proceedings of the 2006 International Semantic Web Conference (ISWC'06)*, Volume 4273 of *Lecture Notes in Computer Science*, pp. 501–514. Springer.

Pan, Z. et J. Heflin (2003). Dldb : Extending relational databases to support semantic web queries. In *Proceedings of the 1st International Workshop on Practical and Scalable Semantic Systems (PSSS'03)*, pp. 109–113.

Park, M. J., J. H. Lee, C. H. Lee, J. Lin, O. Serres, et C. W. Chung (2007). An efficient and scalable management of ontology. In *Proceedings of the 12th International Conference on Database Systems for Advanced Applications (DASFAA'07)*, Volume 4443 of *Lecture Notes in Computer Science*. Springer.

Patel-Schneider, P. F. et I. Horrocks (2006). A comparison of two modelling paradigms in the semantic web. In *Proceedings of the Fifteenth International World Wide Web Conference (WWW'06)*, pp. 3–12. ACM.

Pierra, G. (2003). Context-explication in conceptual ontologies : The PLIB approach. In *Proc. of Concurrent Engineering (CE'2003)*, pp. 243–254.

Spyns, P., R. Meersman, et M. Jarrar (2002). Data modelling versus ontology engineering. Volume 31, New York, NY, USA, pp. 12–17. ACM Press.

Stoffel, K., M. Taylor, et J. Hendler (1997). Efficient management of very large ontologies. In *Proceedings of the 14th National Conference on Artificial Intelligence and 9th Innovative Applications of Artificial Intelligence Conference AAAI'97/IAAI'97*, pp. 442–447.

Sugumaran, V. et V. C. Storey (2006). The role of domain ontologies in database design : An ontology management and conceptual modeling environment. Volume 31, New York, NY, USA, pp. 1064–1094. ACM Press.

Summary

The ANSI/SPARC database architecture has mainly been defined for providing access to data independently of their physical representation. When designing a database according to this architecture, a conceptual model is transformed into a logical model. During this transformation, a large amount of semantics of the data can be lost. As a consequence exchanging/integrating data of different databases or generating user interfaces for data access become difficult. As a model allowing to make explicit the semantics of data, ontologies seem an interesting solution to these problems. In this paper, we show the need to extend the ANSI/SPARC architecture with the ontological level allowing to store ontologies which describe the semantics of data contained in a database. Notice that this extension will not affect existing applications designed according to the initial architecture. First, we define the requirements for exploiting the proposed architecture ; then, we discuss its implementation.