Conception de bases de données à partir d'ontologies de domaine :

Application aux bases de données du domaine technique

Hondjack Dehainsala*, Guy Pierra, Ladjel Bellatreche**, Yamine Aït Ameur****

** Orange FT Group*
hondjack.dehainsala@orange-ftgroup.com
** LISI/ENSMA Université de Poitiers*
1, Avenue Clément ADER, 86960 Futuroscope
(pierra,bellatreche,yamine)@ensma.fr

RÉSUMÉ. *Les méthodologies de conception de bases de données passent par des processus complexes obligeant les concepteurs à connaître le cahier des charges et les différents acteurs de l'application à informatiser. En conséquence, pour un même cahier des charges, deux concepteurs peuvent obtenir deux modèles de bases de données différents. En conséquence, le processus d'intégration de ces bases de données devient complexe. Les ontologies ont été largement utilisées pour automatiser le processus d'intégration de sources de données hétérogènes. Dans cet article, nous identifions d'abord les similitudes et les divergences entre les ontologies de domaine et les modèles conceptuels. Nous proposons ensuite une méthodologie de conception de bases de données basée sur les ontologies. Par rapport aux approches de conception existantes, cette méthodologie offre deux avantages : (1) elle permet de redéfinir la* ***structure des classes*** *en fonction des besoins des applications cibles, et (2) d'importer un ensemble de propriétés de l'ontologie de domaine que le concepteur considère essentiel ou utile pour son application. Ainsi, les bases de données résultantes possèdent leurs propres ontologies locales articulées avec une ou plusieurs ontologies de domaine et peuvent être intégrées d'une manière automatique dans le cas où elles sont candidates dans un processus d'intégration. Finalement, notre méthodologie a été mise en œuvre dans un outil nommé PLIBEditor.*

ABSTRACT.

MOTS-CLÉS : *Conception de base de données, ontologie de domaine, articulation d'ontologies, PLIB, intégration*

KEYWORDS: *database design, ontology, articulation of ontologies, PLIB, Integration*

1. Introduction

Depuis une trentaine d'années, les bases de données se sont développées pour être aujourd'hui au coeur des systèmes d'information des entreprises. La conception de bases de données est réalisée par des méthodologies offrant un cadre de développement bien maîtrisé. Généralement ces méthodologies suivent trois phases essentielles : (1) la phase de conception dans laquelle le concepteur de la base de données répertorie tous les concepts et les relations entre concepts. Cette identification est réalisée à partir du cahier des charges de l'application et/ou des interviews réalisées avec les acteurs de l'application. Le résultat de cette phase est le modèle conceptuel obtenu par l'utilisation d'un formalisme comme le modèle de Chen (entité/association) ou un diagramme de classes d'UML. (2) La phase logique qui permet de traduire le modèle conceptuel en un modèle logique correspondant à un SGBD cible (relationnel, relationnel objet, etc.). Cette traduction est réalisée à l'aide d'un ensemble de règles. (3) La phase physique qui implémente le modèle logique et définit l'ensemble des structures d'optimisation de requêtes.

Bien que ces méthodologies soient largement utilisées, elles présentent des limites : (1) absence de maîtrise du domaine de discours par le concepteur, (2) un modèle conceptuel pour chaque cahier des charges et (3) écart entre le modèle conceptuel et le modèle logique.

– L'absence de maîtrise du domaine de discours par le concepteur : la conception du modèle conceptuel se fait de manière informelle. Elle est réalisée par des interviews et analyse de documents afin d'identifier les concepts et relations pertinents pour l'application à informatiser. Il faut ajouter à cela le temps consacré à l'identification de tous ces concepts.

– Un modèle conceptuel pour chaque cahier des charges : un modèle conceptuel d'un univers donné est spécifié par des questions précises auxquelles on souhaite que le système d'information puisse répondre. En d'autres termes, le modèle conceptuel dépend étroitement de l'objectif applicatif du système en cours de développement. Cela aboutit donc à des modèles conceptuels qui sont différents, même s'ils concernent essentiellement le même domaine et, sans qu'il soit possible d'identifier par des moyens automatiques, des concepts identiques et des concepts différents. En conséquence, les bases de données obtenues à partir des ces modèles conceptuels sont alors hétérogènes et occasionnent de nombreux problèmes lors de toute tentative d'intégration [BEL 06, WAC 01]. Ces problèmes résident essentiellement dans les différents conflits [KIM 91]. On peut citer ainsi les conflits de noms, les conflits de structure, les conflits d'unité de mesures, les conflits de contexte, etc. Ces conflits résultent du fait qu'un modèle conceptuel est considéré comme une étape intermédiaire dans la définition de la base de données, elle-même structurée en fonction des objectifs applicatifs du système cible. Le modèle conceptuel contient donc exclusivement ce qu'il apparaît pertinent au concepteur de représenter effectivement et selon une structure qui dépend fortement des spécificités du concepteur.

Deux modèles conceptuels conçus par deux concepteurs différents pour deux systèmes visant à remplir la même fonction dans le même domaine sont donc soit (1) partiellement différents du point de vue du domaine exact modélisé, soit (2) très différents du point de vue de la structure du modèle résultant. Les modèles logiques générés sont alors encore différents.

– La distance entre les modèles conceptuels et le modèle logique des données : un des objectifs essentiel de la modélisation conceptuelle est de permettre le développement d'un modèle logique de données qui permet la création d'une base de données. Le passage du modèle conceptuel au modèle logique nécessite des opérations de traduction plus ou moins complexes suivant le formalisme de modélisation utilisé (E/A ou objet) et le système de base de données cible (relationnel, relationnel objet ou objet). Le modèle logique résultant de cette traduction est alors différent du modèle conceptuel initial. Les entités, les associations, les spécialisations sont décomposées en de multiples structures de données propres au SGBD cible. Une grande partie de la sémantique portée par le modèle conceptuel (par exemple, les rôles, les cardinalités, la spécialisation, etc. dans les SGBDs relationnels) disparaît.

Face à ces limites, il y a un réel besoin de proposer de nouvelles méthodologies qui permettent de définir des modèles conceptuels ayant plus de place et d'autonomie dans le processus de conception de base de données en offrant la possibilité de le représenter effectivement dans la base de données qu'il décrit. Pour atteindre cet objectif, les ontologies ont été utilisées ces dernières années [NAV 05, SUG 06]. Différents travaux [SUT 93, MIC 94] ont permis de faire abstraction du modèle logique au moyen de programmes qui accèdent directement aux modèles conceptuels. Ce modèle conceptuel représenté dans la base de données, peut être exploité, d'une part, par une API qui permet de cacher la complexité du modèle logique et, d'autre part, par une interface graphique qui permet d'accéder aux données au niveau sémantique. Pour cela, la nécessité de trouver une représentation des modèles conceptuels dans la base de données pour devenir accessibles, et d'établir une liaison entre le modèle logique et le modèle conceptuel représentés dans la base de données, est apparue. Dans ces travaux, la représentation formelle des modèles conceptuels n'est pas accessible aux utilisateurs.

Dans cet article, nous présentons une nouvelle approche de conception de bases de données fondée sur l'utilisation d'ontologies. Cet article comprend 6 sections. La section 2 présente une comparaison entre les ontologies et les modèles conceptuels. La section 3 présente les différentes approches existantes de conception de bases de données à partir d'ontologies et leurs limitations. Notre approche de conception de bases de données en utilisant les ontologies est ensuite présentée. Une implémentation de notre approche est proposée en section 5. Enfin, la section 6 donne une conclusion et récapitule les principaux résultats et suggère quelques perspectives.

. Ontologies et la conception des bases de données

Ces dernières années ont vu l'apparition de travaux qui proposent de nouvelles méthodologies de conception de bases de données fondées sur l'exploitation des ontologies. Ces méthodologies visent à juste titre à tirer profit des ontologies qui ont montré leur efficacité dans bien d'autres domaines en informatique (IA, Web sémantique, intégration de données, e-commerce , base de données à base ontologique [DEH 07]). Avant de présenter ces différentes approches de conception de bases de données à partir d'ontologies, nous faisons un bref aperçu des ontologies et une comparaison entre ontologies et modèles conceptuels.

2.1. *Un bref aperçu des ontologies*

Une ontologie permet de définir de façon *formelle*, *explicite*, *référençable* et *consensuelle* l'ensemble des concepts partagés d'un domaine [GRU 93].

– Par *formelle* nous entendons que l'ontologie est définie dans un langage traitable par machine.

– *Consensuelle* signifie que l'ontologie est acceptée par les membres d'une certaine communauté travaillant dans le domaine de discours. Tous les concepts partagés peuvent être modélisés à l'identique par tous les membres de cette communauté, qui peut être, avec les ontologies normalisées, l'ensemble d'un secteur professionnel.

– *Référençable* désigne la possibilité de référencer de manière unique les concepts ontologiques sans importation des définitions de ces derniers.

– Enfin *explicite* indique que le domaine modélisé est décrit indépendamment d'un point de vue ou d'un contexte implicite. Plusieurs points de vues, légèrement différents pourront donc partager les mêmes concepts.

Ces différentes caractéristiques offrent aux ontologies un haut niveau d'abstraction et la possibilité d'être partagées. Les ontologies sont définies au moyen de langages de définition comme RDF Schéma [LAS 99], DAML+OIL [CON 01], OWL [DEA 04], PLIB [PIE 98], etc.

Formellement, une ontologie peut être définie par un quadruplet :
$O :< C, P, Sub, Applic >$ [BEL 04, PIE 03] avec :

– C : ensemble des classes utilisées pour décrire les concepts d'un domaine donné. Chaque classe est associée à un identifiant universel globalement unique ;

– P : ensemble des propriétés utilisées pour décrire les instances de l'ensemble des classes C. Nous supposons que P définit toutes les propriétés consensuelles dans le domaine. Chaque propriété est associée à un identifiant globalement unique ;

– Sub est la relation de subsomption de signature Sub : C$\rightarrow$ 2^C, qui à chaque classe c_i de l'ontologie, associe ses classes subsumées directes. La relation de subsomption utilisée dans toutes les ontologies est la relation *OOSub* (*is-a*) qui est la relation de subsomption avec héritage. Elle est seulement utilisée entre classes d'une

même ontologie et permet de définir des hiérarchies simples.

– $Applic$ est une fonction de signature $Applic$: C$\rightarrow$ 2^P, qui associe à chaque classe de l'ontologie les propriétés qui sont applicables pour chaque instance de cette classe. Seules les propriétés ayant pour domaine une classe ou l'une de ses super-classes peuvent être applicables pour décrire les instances de cette classe.

A partir de cette formalisation, chaque langage de définition d'ontologies propose de nouvelles constructions [FAN 07]. Nous montrons notamment dans la section 4 que le modèle d'ontologie PLIB définit la relation de subsomption *OntoSub* permettant de réaliser de la subsomption sans héritage.

2.2. *Comparaison entre modèles conceptuels et ontologies*

Pour illustrer la contribution des ontologies dans la conception de bases de données, nous comparons les modèles conceptuels et les modèles d'ontologies. Les ontologies et les modèles conceptuels présentent à la fois des similitudes et des différences [SPY 02, HON 07].

2.2.1. *Similitude*

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.

2.2.2. *Différences*

On peut identifier cinq différences majeures entre les ontologies et les modèles conceptuels.

– *Objectif de modélisation.* Les MCs *prescrivent* l'information qu'ils représentent dans un système informatique particulier. Par exemple, dans un MC_1 destiné à un fournisseur de mémoire, on trouve nécessaire de décrire un disque par sa capacité, son poids, etc. et dans un MC_2 pour un autre fournisseur, on ne devra pas représenter le poids mais le prix de vente. 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. Dans l'exemple précédent, mais en restant toujours dans un contexte d'une ontologie des mémoires, un disque sera caractérisé (directement ou dans une de ses sous-classes) par toutes les propriétés de MC_1 et MC_2 et probablement bien d'autres. Chaque système particulier pourra alors puiser celles des propriétés qui sont pertinentes.

– *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.

.

– *Raisonnement.* Le caractère *formel* des ontologies permet d'appliquer aux ontologies des opérations de raisonnement [BAA 03] soit pour vérifier la cohérence des informations, soit pour en déduire de l'information. Par exemple dans la plupart des modèles d'ontologies (OWL, PLIB) pour une ontologie et une classe données, on peut calculer (1) toutes ses super-classes (directes ou non), (2) ses sous-classes (directes ou non), (3) ses propriétés caractéristiques (héritées ou locales), (4) toutes ses instances (polymorphes ou locales), etc.

– *Consensualité.* Les ontologies étant par définition consensuelles, cela suppose donc que l'ensemble des concepts qu'elles décrivent ont fait l'objet d'un consensus par des acteurs d'une ou de plusieurs "communautés".

– *Souplesse de description.* Les ontologies offrent une certaine souplesse par rapport aux modèles conceptuels dans la description des instances des concepts. On peut noter le fait que les instances des classes d'une ontologie n'ont pas forcément la même structure, i.e., elles peuvent ne pas initialiser les mêmes propriétés. Cette souplesse a pour conséquence de rendre les ontologies beaucoup plus simples à utiliser pour des échanges ou intégrations de systèmes d'informations [BEL 06].

Notons que les ontologies existent de plus en plus dans différents domaines, plus particulièrement dans le domaine technique (ingénierie, CAO, etc.). Les utiliser dans le processus de conception de bases de données constitue un nouveau défi pour la communauté de bases de données.

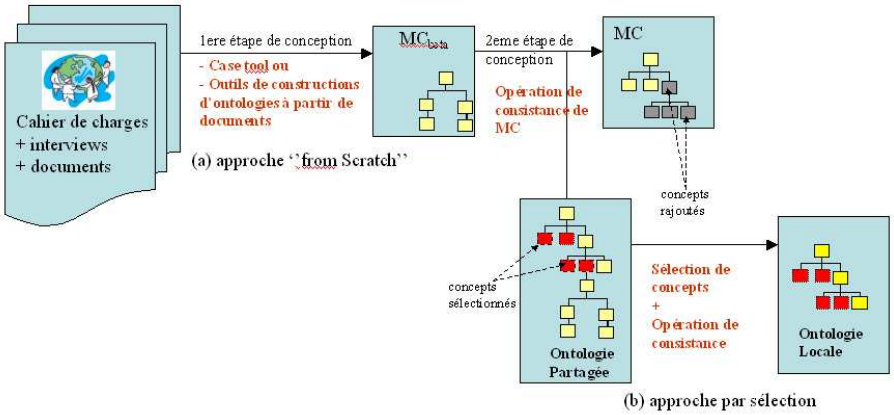

Figure 1. *Approches de conception de bases de données à partir d'une ontologie de domaine*

3. Les approches de conception de bases de données à partir d'ontologies

Si nous nous penchons sur la littérature, nous trouvons deux approches principales utilisant les ontologies dans la conception de bases de données : approches a posteriori

et les approches a priori [NAV 05, SUG 06].

– Les approches a posteriori se déroulent en deux étapes [NAV 05]. Dans la première étape, le concepteur construit un premier modèle conceptuel directement soit à l'aide d'un outil de modélisation classique[1] soit au moyen d'un outil de construction d'ontologies à partir des documents textuels du cahier de charges de l'application cible [2]. Dans la seconde étape, le modèle conceptuel, ainsi constitué, est ensuite mis en correspondance avec une ou plusieurs ontologie(s) du domaine d'application de la base de données cible. Cette opération vise à déceler les concepts et/ou relations et/ou propriétés du modèle conceptuel manquants ou superflus. Cette étape se déroule de façon interactive avec le concepteur où des concepts et/ou des relations et/ou propriétés à supprimer ou à rajouter lui sont proposés. Chaque intervention du concepteur est suivie d'une vérification globale du modèle conceptuel pour assurer la *consistance* par rapport à l'ontologie.

– L'approche a priori permet aux concepteurs d'extraire directement des ontologies de domaine des modèles conceptuels [SUG 06]. Dans ce cas, un modèle conceptuel peut être représenté comme un fragment de l'ontologie. Les concepts, relations et propriétés de la future base de données sont directement sélectionnés à partir d'un ou plusieurs ontologie(s) de domaine pour former une ontologie locale qui sert ensuite à générer le modèle conceptuel de la future base de données.

Afin d'assurer la consistance et la cohérence de l'ensemble des concepts, un ensemble de règles de complétude est défini.

– *Règles de complétude.* Extraction pour une classe ou propriété ou relation donnée de tous les concepts ou propriétés ou relation dont elle dépend. Par exemple, pour une classe donnée, il est nécessaire de disposer dans l'ontologie locale de sa ou de ses super-classes et pour une propriété de son domaine et de son co-domaine.

– **Règles de nettoyage.** Suppression des classes, propriétés, relations superflues ou non utilisées dans l'ontologie locale : par exemple, deux classes équivalentes sélectionnées.

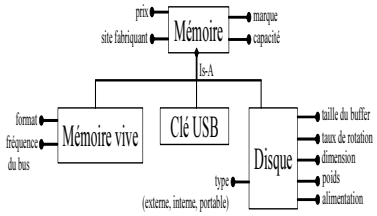

Figure 2. *Un exemple d'un modèle conceptuel*

1. Par exemple PowerAMC
2. Par exemple : ACACIA (http ://www.inria.fr/rapportsactivite/RA2005/acacia/acacia_tf.html)

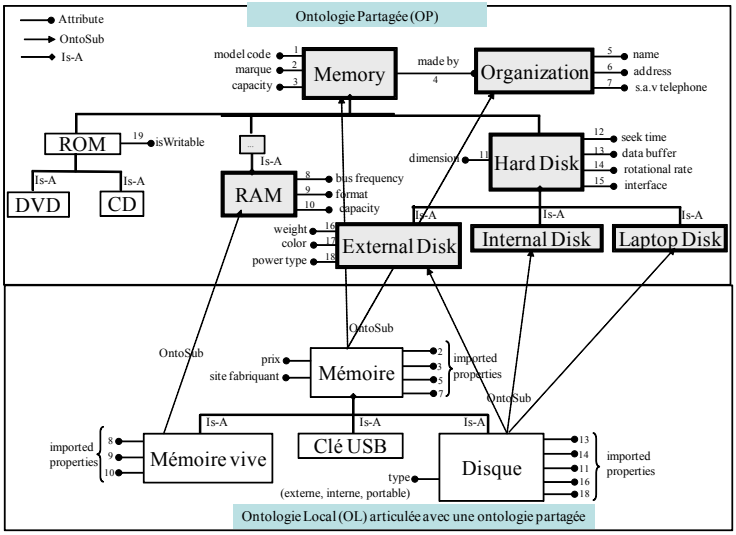

Figure 3. *Extraction de MC et utilisation de la relation de subsomption OntoSub du modèle d'ontologie PLIB*

3.1. *Limites des deux approches*

Bien que ces deux approches permettent de répondre aux limites de l'approche classique de conception de bases de données, elles présentent deux inconvénients majeurs.

– **Extraction de concepts superflus.**Le premier inconvénient est lié à l'opération qui vise à rendre *consistante* les sous-ontologies extraites. En effet, lorsqu'on cherche à rendre *consistante* les sous-ontologies extraites des ontologies, on arrive à extraire de nombreux concepts/relations/propriétés pas forcement désirés pour les besoins de l'application cible. L'extraction du modèle conceptuel de la figure 2 à partir de l'ontologie partagée de la figure 3, nécessiterait (1) l'extraction de la classe *Organization* dû au fait qu'elle est le co-domaine de la propriété *made by* défini dans la classe *Memory* et (2) l'extraction de toutes les classes de la hiérarchie de la classe *RAM* dû au fait que l'extraction de la *RAM* nécessite l'extraction de sa super-classe et ainsi de suite jusqu'à la classe racine (*Memory*). La classe *Organization* et les super-classes de la classe *RAM* sont donc superflues dans la sous-ontologie ainsi construite.

– **Liberté de structuration.** Le second problème est lié à la liberté de structuration de la structure de son modèle de conceptuel qui est fortement dépendante de celle de l'ontologie. En effet, vu que le concepteur sélectionne un sous-ensemble de l'ontologie, celle-ci impose sa hiérarchie de classes. Dans l'exemple de la figure 3, l'ensemble des classes grisées forment la sous-ontologie englobant le modèle conceptuel de la figure 2. La structure de l'ontologie locale est imposée par celle de l'ontologie partagée.

C'est précisément pour résoudre ces deux problèmes que nous proposons une nouvelle approche de conception qui s'applique aux domaines techniques, à savoir le

commerce électronique et l'échange de données techniques dans le domaine de composants industriels, où la notion d'interchangeabilité et de normalisation sont très développées, un vocabulaire technique consensuel (disons une ontologie) existe pour les termes essentiels de chaque domaine. L'intégration de données dans ce domaine présente néanmoins deux difficultés : (1) ces vocabulaires ne couvrent pas les innovations qui apparaissent de façon continue, et (2) chaque base de données, fournisseur ou utilisateur, et chaque catalogue électronique même s'il référence le vocabulaire commun, utilise une structure et une terminologie qui lui est propre. L'imposition d'un vocabulaire ainsi que sa structure, comme le propose les deux approches présentées, sont très contraignantes pour les fournisseurs ou les utilisateurs qui sont amenés à concevoir des bases de données. Dans la section suivante, nous présentons l'approche que nous proposons pour remédier à ces deux inconvénients.

4. Notre approche de conception de bases de données avec ontologies

Notre méthodologie de conception de bases de données suit la seconde approche de construction de base de données à partir d'ontologie qui permet au concepteur de sélectionner les concepts qu'il souhaite intégrer dans son modèle conceptuel. Cette approche présente l'avantage d'offrir plus de contrôle du modèle conceptuel par les concepteurs. Pour ce faire, elle propose une relation de subsomption *OntoSub* (*case of*) sans héritage définie dans le modèle d'ontologie PLIB [PIE 03] qui permet :

– de redéfinir entièrement la structure de classes en fonction des besoins ;

– d'importer des propriétés faisant partie de l'ontologie de façon (1) à pouvoir répondre à des requêtes exprimées en termes de l'ontologie et de fournir une vue ontologique, et (2) à pouvoir exporter le contenu de la base de données correspondant à des domaines couverts par l'ontologie sous forme d'individus de l'ontologie.

4.1. *La relation de subsomption OntoSub du modèle d'ontologie PLIB*

La relation sémantique *OntoSub*, permet de spécifier qu'une classe particulière, est une "sorte" d'une ou plusieurs classes et partage un *sous-ensemble* des propriétés de ces classes. Cette relation est différente de la relation *OOSub* (*is-a*) dans le sens où la classe subsumante importe *explicitement* les propriétés dont elle a besoin de la classe subsumée. La relation *OntoSub* permet de se libérer de la contrainte forte de la relation de subsomption *OOSub* qui permet de spécialiser des classes et impose une importation implicite de *toutes* les propriétés des super-classes.

La relation *OntoSub* est particulièrement intéressante dans le sens où elle offre la possibilité à chaque ontologie locale des sources de données de s'exprimer indépendamment de l'ontologie partagée susceptible d'évoluer. Mais en s'articulant par rapport à l'ontologie partagée, l'ontologie locale reste néanmoins disponible pour s'exprimer et/ou répondre en termes de l'ontologie partagée dans un certain contexte(par exemple d'échanges (ou intégration) de données).

.

Dans l'exemple de la figure 3b, on peut remarquer qu'au moyen de la relation *OntoSub*, on a pu construire une ontologie locale conformément au modèle conceptuel tel que souhaité par le concepteur (cf. figure 2). La classe *Mémoire* de l'ontologie locale subsume la classe *Memory* et importe les propriétés d'identifiant 2,3,5,7. La classe *Disk* subsume les classes *External Disk*, *Internal Disk* et *Laptop Disk*.

Une fois l'ontologie locale s'articule avec l'ontologie partagée, l'ontologie locale peut étendre son schéma en ajoutant de nouvelles classes et/ou propriétés, comme le montre la figure 3b, où une nouvelle classe *Clé USB* qui subsume la classe *mémoire* est ajoutée, et deux nouvelles propriétés (*prix* et *site fabriquant*)au niveau de la classe *Mémoire* sont ajoutées.

L'articulation d'une ontologie locale avec une ontologie partagée doit respecter le principe d'engagement sur une ontologie de référence :

– Toute classe locale doit référencer, par la relation de subsomption *OntoSub*, la plus petite classe subsumante existante dans la hiérarchie de référence si ce n'est pas la même que celle de sa propre super-classe ; elle ne référence une classe que s'il s'agit d'un concept complètement nouveau pour le domaine,

– Toute propriété nécessaire à l'ontologie locale et existant dans l'ontologie de référence doit être importée à travers la relation de subsomption *OntoSub*.

4.2. *Notre méthodologie*

En se basant sur la relation de subsomption *OntoSub*, nous proposons une méthodologie de conception de bases de données qui se déroule en cinq étapes (cf. figure 4).

1) **Construction ou importation d'une ontologie.** Le concepteur d'une base de données doit connaître l'ontologie ou les ontologies de domaine correspondant à son application.

2) **Construction d'une ontologie locale.** Dans cette étape, le concepteur doit identifier les parties pertinentes des ontologies, structurer ces parties en fonction de ses besoins et de son application cible, identifier les relations de subsomptions entre ces catégories, et identifier les plus petites classes subsumantes de ces parties pour appliquer les relations de subsomptions. Enfin, il sélectionne les propriétés applicables dont il a besoin dans les classes subsumantes de son ontologie locale. Notons qu'il est possible qu'aucune ontologie ne couvre de façon suffisamment détaillée le domaine de son application. Dans ce cas, celui-ci peut l'étendre à son problème par l'ajout de nouvelles classes et de nouvelles propriétés.

3) **Construction du modèle conceptuel.** A partir de l'ontologie locale, le concepteur peut désormais construire son modèle conceptuel. Cette étape est réalisée par la sélection des classes et des propriétés que le concepteur souhaite effectivement représenter dans la base de données. Rappelons que cette étape est nécessaire compte tenu du caractère *prescriptif* des modèles conceptuels et du caractère *descriptif* des on-

ologies. L'ontologie locale définie dans l'étape précédente est encore assez générale (on peut supposer que celle-ci n'est qu'une re-structuration de l'ontologie partagée). Le concepteur se doit, une fois de plus, d'identifier les classes et les propriétés qu'il souhaite effectivement représenter. Notons que dans la plupart des cas, le concepteur construit son ontologie locale en fonction des besoins de son application et il sélectionne donc quasiment toutes les classes et propriétés de son ontologie locale.

4) **Définition du modèle logique**. A l'aide d'un ensemble de règles de correspondance choisies, le modèle conceptuel est traduit en modèle logique qui définit la structure des données sous forme interprétable par le SGBD cible.

5) Enfin, le SGBD peut alors générer la représentation physique des données.

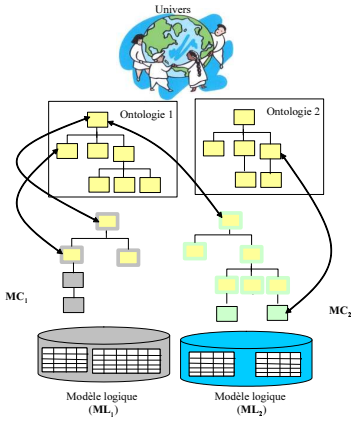

Figure 4. *Méthodologie de conception de bases de données à l'aide d'ontologie*

Dans la section 1, nous avons identifié trois limites dans la méthodologie usuelle de conception de bases de données. Nous montrons comment notre approche permet de résoudre ces limites.

(1) L'absence de maîtrise du domaine d'étude par le concepteur.
Notre proposition. Comme nous envisageons de se fonder sur des ontologies de domaine et que celles-ci sont précises et documentées, elles peuvent constituer un point de départ pour la conception des modèles conceptuels.

(2) Forte dépendance des modèles vis-à-vis des concepteurs et des applications.
Notre proposition. Notre approche permet de conserver cette dépendance à la différence que le modèle conceptuel construit est extrait d'une ontologie locale *structurée* en fonction du besoin du concepteur et *articulée* avec une ou plusieurs ontologies partagées. Ce qui permet d'éviter tous les conflits liés à l'approche classique de conception de bases de données.

(3) Distance entre les modèles conceptuels et le modèle logique de données.
Notre proposition. Notre approche propose toujours de traduire les modèles conceptuels en modèles logiques en appliquant des règles de correspondance entre ces deux univers. Pour éviter cette conséquence de la correspondance, il est alors apparu béné-

.

fique de donner plus de place et d'autonomie à la notion de modèle conceptuel pour permettre sa représentation effective dans une base de données.

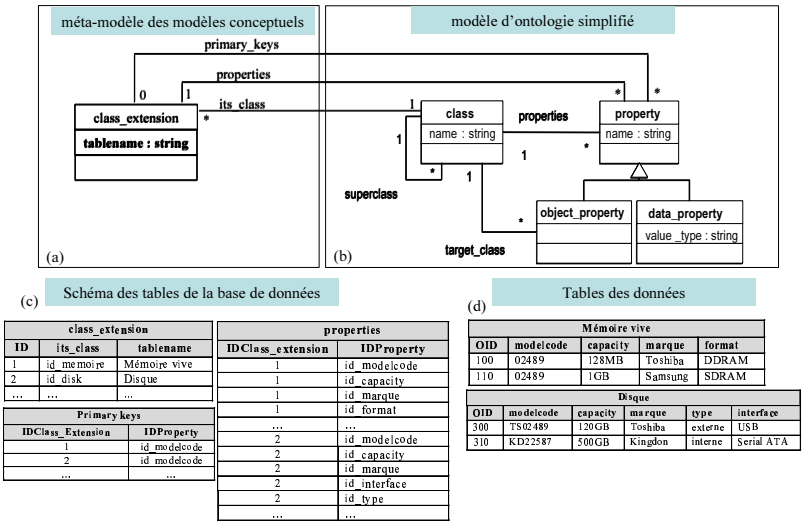

Figure 5. *Exemple de représentation du modèle conceptuel des instances des classes de la base de données*

4.3. *Représentation du modèle conceptuel dans la base de données*

Pour la représentation des modèles conceptuels, nous avons proposé un méta-modèle. Une version simplifiée de ce méta-modèle est présenté dans la figure 5a. Une particularité est à noter sur notre modèle : il fait référence à notre modèle d'ontologie pour la simple raison que, dans notre approche, les modèles conceptuels sont extraits de (ou des) ontologies locales. Dans notre implémentation, nous avons opté pour représenter également, au sein de la base de données, les ontologies locales. La figure 5b présente une vue simplifiée d'un modèle d'ontologie. Notons que la représentation des ontologies dans la base de données n'est qu'un choix d'implémentation. Cette représentation offre toutefois l'avantage de rendre disponible dans la base de données plus de sémantique car décrite dans les ontologies peuvent être exploitées lors d'une intégration future ou pour la génération d'IHM ou de programmes.

La figure 5a présente un diagramme de classes UML décrivant un modèle simplifié de modèle conceptuel. Dans cette représentation, on peut remarquer que chaque extension de classe référence sa classe à travers l'attribut *its_class*. L'ensemble des propriétés initialisées par la classe est donné par la relation *properties*. La relation *primarykeys* permet de spécifier les propriétés de la classe qui forment la clé pour identifier une instance. La figure 5c montre un schéma de tables issu du diagramme UML et contenant un modèle conceptuel sélectionné de l'ontologie locale (cf. figure

3b). Enfin la figure 5d montre un schéma de tables des classes *Mémoire vive* et *Disque* du modèle conceptuel. Ces tables sont peuplées avec un ensemble de données.

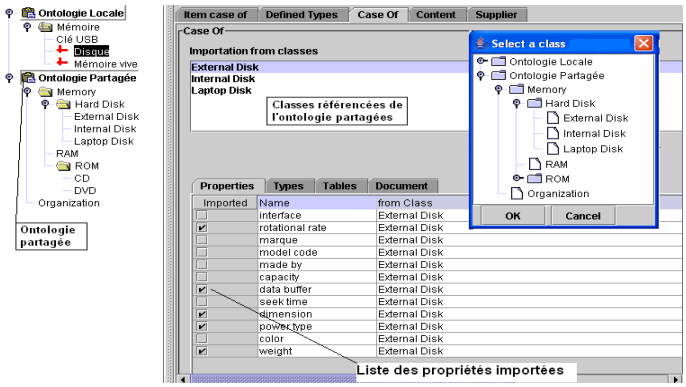

Figure 6. *Construction d'une ontologie locale avec PLIBEditor*

5. Implémentation de notre approche

Notre proposition a été implémentée dans l'environnement PLIBEditor, un outil de conception d'ontologies PLIB. Sur ce dernier, les étapes de conception de bases de données ont été implémentées. La figure 6 présente l'interface permettant l'articulation d'une classe (*Disque*) par rapport aux classes (*External Disk*, *Internal Disk* et *Laptop Disk*) d'une ontologie partagée. La zone centrale de la fenêtre permet d'importer (ou de sélectionner) les propriétés de la classe subsumée dans la classe subsumante. La figure 7 présente l'interface permettant la construction du schéma des tables de la base de données. Le concepteur, par cette interface, sélectionne les classes puis les propriétés de l'ontologie locale qu'il souhaite intégrer dans son modèle conceptuel. Le modèle conceptuel ainsi conçu est automatiquement stocké dans une base de données et les tables et colonnes correspondant aux classes et propriétés sont automatiquement créées dans celle-ci. Ces tables de données peuvent donc être modifiés dynamiquement et être peuplés directement à travers l'interface de PLIBEditor.

PLIBEditor permet de se connecter directement à une base de données PostgreSQL dans laquelle sont stockées à la fois les ontologies partagées, l'ontologie locale, le modèle conceptuel et les tables de données. Dans la version actuelle de *PLIBEditor*, les modèles conceptuels sont définis interactivement. Nous envisageons dans une prochaine version, d'offrir la possibilité de concevoir directement avec un formalisme de modélisation (UML, OMT ou Merise), puis ensuite de le répercuter dans la base de données.

.

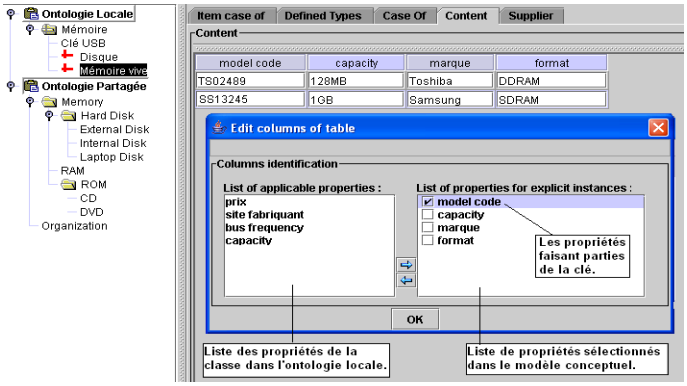

Figure 7. *Sélection des propriétés d'ontologie pour constituer le MC.*

6. Conclusion

Dans ce travail, nous avons proposé une nouvelle approche de conception de bases de données fondée sur les ontologies de domaine. Cette approche, à la différence des approches existantes, vise à utiliser la relation de subsomption sans héritage du modèle d'ontologie PLIB (ISO 13584) qui offre l'avantage de concevoir des sources de données autonomes et disposant de leur propre ontologie locale articulée par rapport à une ou plusieurs ontologie(s) partagée(s) pour pouvoir répondre à des requêtes par rapport à celle(s)-ci.

Notre méthodologie s'applique au domaine technique où de nombreuses ontologies se construisent de plus en plus et sont partagées entre de nombreux fournisseurs industriels. Notre méthodologie est constituée de cinq étapes. La première étape vise à rechercher des ontologies qui couvrent le domaine des applications visés par la base de données à définir. Dans la deuxième étape, une ontologie locale est construite par extraction des concepts et propriétés des ontologies partagées sélectionnées dans l'étape précédente. Le concepteur dispose d'une liberté totale dans la structuration des classes et des propriétés qui l'intéressent. Dans la troisième étape, le modèle conceptuel de l'application est défini à partir de l'ontologie locale. Une fois de plus un sous-ensemble de classes et de propriétés est désigné pour faire partie du modèle conceptuel. Notons que cette étape est justifiée par le caractère *descriptif* des ontologies et la nature *prescriptif* des modèles conceptuels. Mais, dans la plupart des cas, le concepteur choisit toutes les classes et propriétés sélectionnées dans la deuxième étape. Dans la quatrième étape, le modèle conceptuel est traduit en modèle logique à partir d'un ensemble de règles de transformation. Enfin la cinquième étape où le modèle logique est converti en modèle physique propre à un SGBD.

Afin d'apporter plus de sémantique aux sources de données, nous avons opté pour la représentation au sein des sources de données à la fois des ontologies locales et des modèles conceptuels et éventuellement des ontologies partagées. Cette représentation permet justement de réduire la distance en les modèles conceptuels et les mo-

dèles logiques résultant issus de transformation. Au moyen de programmes, les ontologies et les modèles conceptuels dans la base de données peuvent être exploités au moyen d'une API et permettre, entre autres, la génération de programmes et d'interface homme machine. La méthodologie que nous avons proposée a été mise en œuvre à travers l'outil PLIBEditor dont des démonstrations "flash" et des captures d'écran sont disponibles à cette adresse : *http ://www.plib.ensma.fr/plib/demos/ontodb/ index.html*.

Comme perspectives à ce travail, nous envisageons d'étudier l'expressivité de la relation *OntoSub* dans d'autres modèles d'ontologies tels que OWL ou FLIGHT. Nous envisageons également de représenter au sein des bases de données (1) **les dépendances fonctionnelles [ontologiques] extensionnelles** entre les propriétés des classes, qui nous permettrons par la suite d'optimiser les schémas logiques de données en les normalisant (i.e., 1FN, 2FN, 3FN) ; (2) **les dépendances fonctionnelles [ontologiques] intensionnelles** pour exprimer les relations ou fonctions (au sens mathématiques) existantes entre les propriétés des classes et qui nous permettrons de calculer les valeurs des unes par rapport aux autres dynamiquement ; (3) **les contraintes d'intégrité [ontologique]** pour exprimer des relations logiques entre les propriétés qui doivent être vérifiées (i.e., toujours vraies) pour toutes les instances des classes (invariant de classe).

7. Bibliographie

[BAA 03] BAADER F., CALVANESE D., MCGUINNESS D. L., NARDI D., PATEL-SCHNEIDER P. F., Eds., *The Description Logic Handbook : Theory, Implementation, and Applications*, Cambridge University Press, 2003.

[BEL 04] BELLATRECHE L., PIERRA G., XUAN D., HONDJACK D., « Intégration de sources de données autonomes par articulation a priori d'ontologies », *Proc. du 23ème congrès Inforsid*, may 2004, p. 283-298.

[BEL 06] BELLATRECHE L., XUAN D., PIERRA G., HONDJACK D., « Contribution of Ontology-based Data Modeling to Automatic Integration of Electronic Catalogues within Engineering Databases », *Computers in Industry Journal*, vol. 57, n° 8-9, 2006.

[CON 01] CONNOLLY D., HORROCKS I., MCGUINNESS D., PATEL-SCHNEIDER F., STEIN A., « DAML+OIL Reference Description », *World Wide Web Consortium*, , 2001.

[DEA 04] DEAN M., SCHREIBER, « WL Web Ontology Language Reference », *W3C Recommendation (2004)*, , 2004.

[DEH 07] DEHAINSALA H., PIERRA G., BELLATRECHE L., « OntoDB : An Ontology-Based Database for Data Intensive Applications », *in the 12th International Conference on Database Systems for Advanced Applications (DASFAA'07), edited by Springer's Lecture Notes in Computer Science*, Bangkok - Thailand, April 2007.

[FAN 07] FANKAM C., AIT-AMEUR Y., PIERRA G., « Exploitation of ontology languages for both persistence and reasoning purposes : Mapping PLIB, OWL and Flight ontology models », *in Third International Conference on Web Information Systems and Technologies(WEBIST), INSTICC Press*, 2007.

[GRU 93] GRUBER T., « A translation approach to portable ontology specification », *Knowledge Acquisition*, vol. 7, 1993.

[HON 07] HONDJACK D., « Explicitation de la sémantique dans les bases de données : base de données à base ontologique et le modèle OntoDB », *Thèse de Doctorat Université de Poitiers*, , 2007.

[KIM 91] KIM W., SEO J., « Classifying Schematic and Data Heterogeneity in Multidatabase Systems », *Computer*, vol. 24, n° 12, 1991, p. 12–18, IEEE Computer Society Press.

[LAS 99] LASSILA O., SWICK R., « Resource Description Framework (RDF) Model and Syntax Specification », *World Wide Web Consortium*, , 1999.

[MIC 94] MICHAEL R. BLAHA WILLIAM J. PREMERLANI H. S., « Converting OO Models Into RDBMS Schema », *IEEE Software*, vol. 11, n° 3, 1994, p. 28-39.

[NAV 05] NAVAS-DELGADO I., ALDANA-MONTES J. F., « A Design Methodology for Semantic Web Database-Based Systems », *ICITA '05 : Proceedings of the Third International Conference on Information Technology and Applications (ICITA'05) Volume 2*, Washington, DC, USA, 2005, IEEE Computer Society, p. 233–237.

[PIE 03] PIERRA G., « Context-Explication in Conceptual Ontologies : The PLIB Approach », *Proc. of Concurrent Engineering (CE'2003)*, July 2003, p. 243-254.

[PIE 98] PIERRA G., POTIER J. C., BATTIER G., SARDET E., DEROUET J. C., WILLMANN N., MAHIR A., « EXCHANGE OF COMPONENT DATA : THE PLIB (ISO 13584) MODEL, STANDARD AND TOOLS », September 98, p. 160-176.

[SPY 02] SPYNS P., MEERSMAN R., JARRAR M., « Data modelling versus ontology engineering », *SIGMOD Rec.*, vol. 31, n° 4, 2002, p. 12–17, ACM Press.

[SUG 06] SUGUMARAN V., STOREY V. C., « The role of domain ontologies in database design : An ontology management and conceptual modeling environment », *ACM Trans. Database Syst.*, vol. 31, n° 3, 2006, p. 1064–1094, ACM Press.

[SUT 93] SUTHERLAND J., POPE M., RUGG K., « The Hybrid Object-Relational Architecture (HORA) : an integration of object-oriented and relational technology », *Proceedings of the 1993 ACM/SIGAPP symposium on Applied computing*, 1993, p. 326–333.

[WAC 01] WACHE H., VÖGELE T., VISSER U., STUCKENSCHMIDT H., SCHUSTER G., NEUMANN H., HÜBNER S., « Ontology-based Integration of Information - A Survey of Existing Approaches », *Proceedings of the International Workshop on Ontologies and Information Sharing*, , 2001, p. 108-117.