Construction d'une ontologie à partir d'une base de données relationnelle : approche dirigée par l'analyse des formulaires HTML

Sidi Mohamed Benslimane*,, Djamal Benslimane*, Mimoun Malki ****, **Youssef Amghar***, **Faiez Gargouri*****

**Laboratoire LIRIS, Université Claude Bernard – INSA Lyon*
8 Bld Niels Bohr, 69622, Villeurbanne Cedex, France
{Sidi-mohamed.benslimane, Djamal.benslimane, Youssef.amghar}@liris.cnrs.fr

*** Laboratoire EEDIS, Université Djillali Liabes de Sidi Bel-Abbes,*
B.P. 89, cité Ben M'Hidi, 22000 Sidi-Bel-Abbes, Algérie
{Benslimane, Malki}@univ-sba.dz

**** ISIM - Sfax*
BP 1030- 3018, Sfax. Tunisie.
faiez.gargouri@fsegs.rnu.tn

RÉSUMÉ. *L'émergence et la généralisation du Web dans tous les domaines, a permis à de nombreuses entreprises d'offrir une variété de services et d'informations en ligne, suscitant ainsi un réel besoin de partage et d'interopérabilité. Cela nécessite une infrastructure permettant à des agents logiciels d'exploiter, de composer et de raisonner sur les contenus constituants les ressources Web. Malheureusement, la notion d'ontologie qui est souvent au cœur de cette infrastructure, n'est pas toujours disponible. Pour contribuer à résoudre ce problème, nous préconisons un processus semi-automatique d'acquisition d'ontologie OWL à partir d'une base de données relationnelle en analysant par des techniques de retro ingénierie les formulaires HTML relatifs à une application Web. L'objectif de cette démarche et de rendre exploitables par les machines, les bases de données relationnelles disponibles sur le Web, réduisant ainsi le coût de construction des ontologies.*

ABSTRACT. *The emergence of the Internet technology and the rapid growth of its applications have made the information available anywhere and anytime. Thus, most businesses run Web-based front-end databases and make a variety of services and information available online. The next generation of the Web, the Semantic Web, seeks to add machine understandable content to Web resources. Such added content is called ontologies. In this paper we present a semi-automatic reverse engineering approach to acquire OWL ontology corresponding to the content of relational database based on the analysis of its related HTML-forms. The main reason for this construction is to make the relational database information that is available on the Web machine-processable and reduce the time consuming task of ontology creation.*

MOTS-CLÉS : *Extraction d'ontologie, bases de données relationnel, OWL, formulaires HTML, Rétro-ingénierie.*

KEYWORDS: *Ontology extraction, Relational databases, OWL, HTML forms, Reverse engineering.*

1. Introduction

L'accroissement des technologies du Web et le développement rapide de ses applications, ont rendu l'information disponible n'importe où et n'importe quand. Ainsi plusieurs entreprises tentent de rendre accessibles sur le Web une variété de leurs services, suscitant ainsi un besoin de partage et d'interopérabilité. La prolifération et la disponibilité des ontologies sont cruciales pour le succès de cette démarche qui s'inscrit dans le cadre du Web sémantique. Néanmoins la construction des ontologies demeure si coûteuse qu'elle entrave le progrès des activités du Web sémantique. La construction manuelle des ontologies (Volz ; *et al.,* 2004) demeure toujours une tâche lourde, longue, et encombrante. La construction automatique des ontologies à partir de sources d'informations existantes (Haustein *et al.,* 2002) par des outils entièrement automatisés est toujours à un stade préliminaire de maturité. Par conséquent, l'utilisation d'un processus de construction semi-automatique d'ontologies est intéressante de ce point de vue et peut être considérée comme une solution intermédiaire et pratique.

Actuellement, les bases de données relationnelles demeurent le moyen le plus populaire pour stocker, rechercher et manipuler des données, cependant, la structure et les contraintes d'intégrité du modèle relationnel sont définies par des schémas qui ne sont pas aussi expressifs que des ontologies, pour ce qui est de la représentation de la sémantique des données. Par conséquence, il est essentiel de construire des ontologies qui soutiennent sémantiquement l'information contenue dans ces bases de données. La technique de rétro ingénierie, semble être une solution intéressante pour atteindre cet objectif. Elle est définie comme un processus d'analyse d'un système permettant l'identification des entités et leurs liens en vue de passer d'une forme de représentation à une autre, de niveau d'abstraction identique ou plus élevé (Chiang *et al.,* 1994). Cependant, les informations extraites à partir d'un schéma relationnel pour la construction d'ontologie peuvent être limitées:

- Pour des raisons de performance, souvent, les concepteurs de base de données peuvent être amenés à ne pas respecter les règles de normalisation pour optimiser le schéma.
- Les schémas ne sont pas toujours en troisième forme normale.
- Les informations complètes sur la base de données relationnelle, telle que des dépendances fonctionnelles et d'inclusion, sont rarement disponibles (Premerlani, *et al.,* 1994).
- Etant donné que le modèle relationnel ne supporte pas tous les constructeurs du modèle conceptuel, une partie de la sémantique capturée dans le schéma conceptuel est nécessairement perdue lors du passage au schéma relationnel (c'est par exemple le cas de l'héritage).
- Les noms des relations et des attributs du schéma relationnel sont souvent abrégés ou ambigües (e.g. CUST_NB, StuName, S_125_AZE, etc). Ainsi, il est difficile ou même impossible de déduire la signification (i.e. la sémantique) des données en se basant sur ces appellations (Muller, 1998).

Dans ce papier nous proposons une approche de construction semi-automatique d'ontologie OWL basée sur l'analyse d'une base de données relationnelle ainsi que les pages HTML associées. Le reste du papier est organisé comme suit : dans la section 2, nous discutons un certain nombre de travaux relatifs à la rétro ingénierie des bases de données relationnelles vers les ontologies. La section 3, présente l'architecture générale de notre système. Les étapes d'extraction du schéma de formulaire sont détaillées dans la section 4. La section 5 décrit le processus d'enrichissement du schéma relationnel, tandis que la section 6 détaille les règles de construction de l'ontologie OWL, Enfin, nous concluons et donnons des perspectives à ce travail dans la section 7.

2. Travaux antérieurs

Les travaux sur la rétro ingénierie de bases de données relationnelles, suggèrent des méthodes et des règles pour définir explicitement la sémantique dans le schéma de base de données (Biskup, 1998), extraire la sémantique d'un schéma de base de données (Chiang et al., 1994) et transformer un modèle relationnel en un modèle orienté objet (Hainaut et al., 1996). Toutefois, la sémantique obtenue par ses approches ne peut pas être employée pour construire aisément une ontologie à partir d'une base de données relationnelle. Bien que le modèle orienté objet soit proche du modèle ontologique, quelques caractéristiques les différencient. Par exemple, la notion d'hiérarchie et de cardinalité **entre propriétés** n'existe pas dans le modèle orienté objet.

Récemment, d'autres approches considérant les ontologies comme le résultat d'un processus de rétro ingénierie des bases de données relationnelles ont été proposées. Ces approches peuvent être classées en trois catégories: *1. Approches basées sur l'analyse des requêtes utilisateurs :* Dans (Kashyap, 1999), l'approche proposée, construit l'ontologie en analysant tout d'abord le schéma relationnel. L'ontologie est ensuite raffinée en utilisant des requêtes d'utilisateurs. A noter que cette approche ne crée pas d'axiomes, qui représentent une partie intégrante d'une ontologie. *2. Approches basées sur l'analyse du schéma relationnel :* Dans (Stojanovic, et al., 2002), l'approche proposée, fournit un ensemble de règles pour transformer les constructeurs de la base de données relationnelle en constructeurs sémantiquement équivalents dans l'ontologie. Ces règles sont basées sur une analyse des relations, des clés et des dépendances d'inclusion. Cependant, l'ontologie construite est exprimée dans RDF(S), langage qui ne possède pas de modèle d'inférence, limitant ainsi les traitements automatiques. *3. Approches basées sur l'analyse des tuples*: Dans (Astrova *et al.,* 2004), l'approche tente d'analyser les tuples de la base de données relationnelle pour découvrir la sémantique "cachée"(e.g. l'héritage). Cependant, cette approche consomme beaucoup de temps au regard du nombre de tuples de la base de données relationnelle.

Pour résoudre les problèmes communs de la rétro ingénierie en vue d'extraire la sémantique à partir des bases de données relationnelles, de nouvelles approches proposent d'analyser les pages HTML. Cette analyse est basée sur la génération de

« wrapper » (Wang, *et al.,* 2003 ; Embley, *et al.*, 2004). Néanmoins, comme le souligne (Florescu, *et al.,* 1998), les pages HTML sont souvent restructurées (en moyenne plus de deux fois par an). Ainsi tout changement dans la structure de la page HTML peut rendre caduque le fonctionnement du « wrapper » et par conséquent les ontologies qui y sont basées. Récemment, (Astrova, *et al.,* 2005) a proposé une approche de construction d'ontologie basée sur l'analyse des formulaires HTML. L'inconvénient de cette approche, est qu'elle ne permet pas l'identification des relations d'héritage dans l'ontologie.

3. Approche pour la construction d'une ontologie

Notre approche pour la construction d'une ontologie, est basée sur l'idée que la sémantique de la base de données relationnelle peut être extraite en analysant les formulaires HTML de l'application Web associée. Cette sémantique sera utilisée pour restructurer et enrichir le schéma relationnel. Un ensemble de règles de transformation permettra de construire directement l'ontologie à partir de ce schéma enrichi. La figure 1 présente l'architecture proposée dans notre approche.

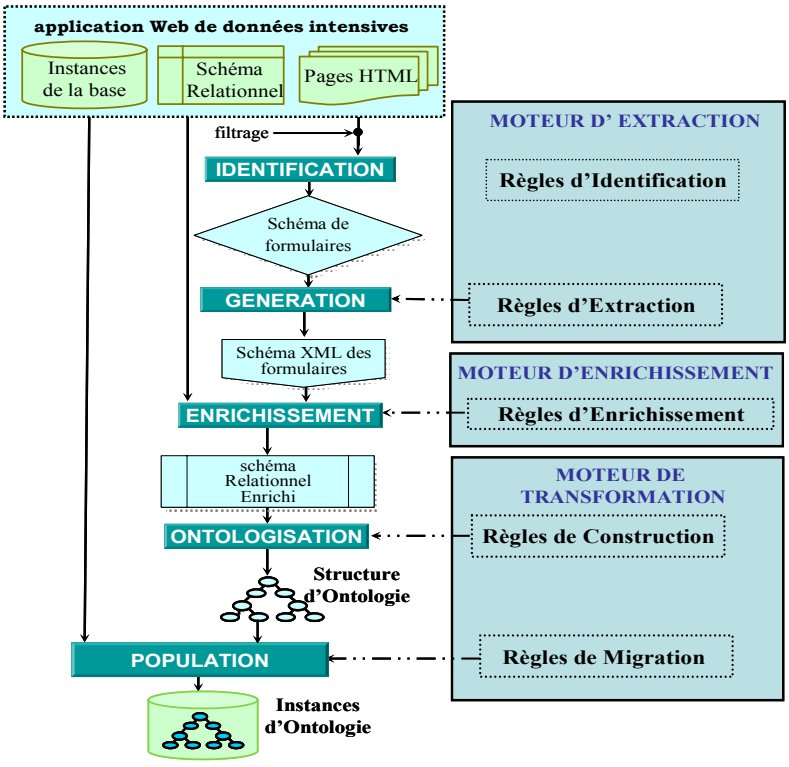

Figure1 : *Architecture générale du système de construction d'ontologie*

Les principaux composants de cette architecture sont :

Le moteur d'extraction : Composé de deux ensembles de règles d'extraction. Le premier ensemble de règles analyse les pages HTML pour identifier les constructeurs du schéma de formulaire. Le second ensemble de règles permet la transformation du schéma de formulaire en schéma XML et la dérivation de la sémantique du domaine par l'extraction du schéma relationnel des formulaires.

Le moteur d'enrichissement : Composé d'un ensemble de règles d'enrichissement, il permet l'intégration de la sémantique extraite à partir du schéma de formulaires dans le schéma relationnel de la base de données.

Le moteur de transformation : Composé de deux ensembles de règles de transformation. Le premier ensemble de règles permet la construction de l'ontologie OWL à partir du schéma relationnel enrichi. Ces règles sont organisées en quatre groupes : i) règles pour construire les classes, ii) règles pour construire les propriétés, iii) règles pour construire les liens d'héritage et iv) règles pour construire les axiomes. Le deuxième ensemble de règles permet la migration des données en créant les instances de l'ontologie à partir des tuples relationnels.

Notre processus d'extraction d'ontologie s'articule autour de trois étapes : l'extraction du schéma de formulaires, l'enrichissement du schéma relationnel de la base de données et la transformation du schéma enrichi en une ontologie OWL.

4. Extraction du schéma de formulaires

Pour illustrer nos propos, nous considérons un site de réservation de vol http://www.airalgerie.dz d'une compagnie aérienne. La figure 2, montre deux pages HTLM parmi les pages de l'application, à savoir le formulaire de réservation (Booking-form) et le tableau de programme de vols (Program of flights). La source de données dont sont issus les données est une base de données relationnelle (voir Table 1). Les attributs soulignés indiquent les clés primaires, alors que les attributs en italiques indiquent les clés étrangères.

Table1. *Schéma de base de données relationnelle*

Passenger	(PassengerID, FN, LN, Age)
City	(CityID)
Departure-City	(*CityID*, DC-Name)
Arrival-City	(*CityID*, AC-Name)
Date	(DeparatueDate)
Hour	(HourID)
Departure-Hour	(HourID, type)
Arrival-Hour	(HourID, type)
Company	(CompagnyID, CompanyName)
Plane	(PlaneID, *CompID*, Capacity)
Leaving-From	(*FlightID, DepartureCityID*)
Going-To	(*FlightID, ArrivalCityID*)
Flight	(FlightID, *Dep-CityID, Arr-CityID, Dep_HourID, Arr_HourID, PlaneID*)
Book	(*PassengerID, FlightID, DepartureDate*, Class)

Figure 2. *: Formulaire et Tableau HTML*

4.1. Analyse de la structure des pages HTML

Le but principal de cette phase est de comprendre et expliciter la signification des formulaires HTML et des tableaux HTML[1], en analysant leur structure et les données qu'ils contiennent pour identifier leurs composants et leurs corrélations.

4.1.1 *Le modèle de forme*

Un formulaire est une collection structurée de champs d'information convenablement formatés pour des données en entrée/sortie d'une base de données. Dans le modèle de formulaire que nous utilisons (Malki, *et al.,* 2002), nous faisons la distinction entre le type et l'instance de formulaire. Ce modèle est similaire mais non identique au modèle présenté dans (Mfourga, 1997). Ce modèle se compose principalement de : *Type de formulaire* : C'est une collection structurée de champs vides composée pour communiquer avec la base de données. Une représentation particulière de type de formulaire s'appelle le *template* de formulaire. *Unités structurelles* : C'est un groupe de champs d'informations homogènes correspondant

[1] Dans ce qui suit, le terme formulaire désigne à la fois les formulaires et les tableaux.

à une zone dans le formulaire HTML. Ces champs d'informations sont généralement rangés en boites composées au minimum : d'un bloc-texte, d'un bloc-graphique, d'un tableau, ou d'un champ d'information. *Instance de formulaire*: C'est une occurrence d'un type de formulaire. La figure 2 illustre deux exemples d'instance. *Champ de formulaire*: C'est un attribut simple qui représente une information de type entier, réel, booléen, texte, etc. Chaque champ doit avoir un nom et une identification unique dans l'ensemble des champs appartenant à un formulaire. Un champ de formulaire est généralement lié à un attribut d'une table dans la base de données. Nous désignerons par "attribut-lié" ce type d'attribut. Un formulaire contient aussi des champs calculés, et des champs qui ne sont pas liés à la base. Nous distinguons trois autres types de champs : les champs de saisie (e.g. *TEXTES, CHECKBOX, RADIO, TEXTAREA*, etc.), les champs de sélection (e.g. *SELECT*) permettant à l'utilisateur de faire un ou plusieurs choix (e.g. *MULTIPLE*), et les Champs de liaison (e.g. *HREF*) utilisés pour relier deux ou plusieurs formulaires (pages). *Source de données* : C'est une structure de la base de données relationnelle, qui définit les relations ainsi que les attributs et leurs types de données. *Liens entre les unités structurelles* : La relation sémantique entre les unités structurelles est définie soit par des liens de hiérarchie, soit par des liens d'association. *Contrainte* : C'est une règle qui définit quelles données sont valides pour un champ de formulaire.

4.1.2 *Règles d'identification du schéma de formulaire*

Les règles décrites ci-dessous récapitulent brièvement le processus utilisé pour identifier le schéma de formulaire. Ces règles font partie du moteur d'extraction.

Règle 1 : Identification des instances de formulaire. Afin de distinguer clairement les différents types d'information, les pages Web sont généralement divisées en plusieurs zones. Chaque zone est créée en utilisant des balises spécifiques. Dans notre approche, nous effectuons un processus de filtrage permettant de considérer deux types de zones : Zones limitées par la balise <forme>, utilisée pour accéder et mettre à jour la base de données. Et zones limitées par les balises (<table>, <td>, <tr>, <li>, <ul>, <thead>, <th>) utilisées pour afficher le résultat d'une requête représentant une vue particulière de la base de données.

Règle 2 : Identification des attributs liés. Un attribut lié peut être identifié en examinant soit : i) pour un formulaire d'entrée, la clause "name" de la balise de saisie <Input> ; ii) pour un formulaire de sortie (tableau), les balises structurelles (<thead>, <th>) (Tijerino, *et al.,* 2005).

Règle 3 : Identification des unités structurelles. Pour déterminer la structure logique d'un formulaire nous utilisons l'approche des sélections visuelles (Yang, *et al.,* 2001). Par exemple l'utilisateur pourrait considérer que les attributs "FirstName", "LastName", et "Age" dans Figure 2 forment un seul groupe "Passenger" en raison de leur apparition dans une même zone du formulaire.

Règle 4 : Identification des liens entre les unités structurelles. Les liens entre deux unités structurelles peuvent être identifiés lorsque les unités sont adjacentes dans la page HTML. De plus les hyperliens peuvent également révéler des liens entre les unités structurelles.

Règle 5 : Extraction de contraintes. En plus des structures des pages HTML, nous analysons également les données contenues dans ces pages pour identifier les contraintes. Une analyse de données inclut une stratégie d'apprentissage par des exemples, empruntée aux machines d'apprentissage (Michalski, 1983). Dans la figure 2, nous identifions la contrainte "Not Null" sur les attributs "Departure-City" et "Arrival-City". Ces attributs contiennent des valeurs non nulles pour toutes les occurrences du formulaire "Booking Form".

4.2 Génération du schéma XML

Une fois le schéma de formulaire identifié, le schéma XML est directement généré en se basant sur des règles de traduction entre les concepts du modèle de formulaires et ceux du schéma XML.

Règle 1: Elément complexe. Chaque unité structurelle est transformée en élément "ComplexType" dans le schéma XML correspondant. Ainsi, l'unité structurelle "Passenger" est traduite par :

<xsd: complexType name="Passenger"> …</xsd: complexType>

Règle 2 : Elément simple. Les champs simples (i.e. non décomposables) sont transformés en sous-éléments de l'élément "ComplexType" correspondant. Leur type primitif est celui du champ. Par exemple, le champ "FirstName" est traduit par :

<xsd: element name="Firstname" type="xsd:string"/>

Règle 3 : Cardinalité minimale. Si l'unité structurelle contient des champs de saisie simple (e.g. la balise *TEXT*), l'élément "ComplexType" correspondant prend comme cardinalité minimal : "minOccurs=1" et maximal : "maxOccurs=1".

Règle 4 : Cardinalité maximale. Si l'unité structurelle contient des champs de saisie multiple (e.g. la balise *MULTPLE*), l'élément "ComplexType" correspondant prend comme cardinalité maximal : "maxOccurs=*".

4.3 Extraction de la structure arborescente du formulaire

Afin de faciliter l'opération d'interprétation et d'extraction de la sémantique des données, nous transformons le schéma XML du formulaire en une structure arborescente selon les étapes suivantes :

1. Définir le noeud racine dont le nom est celui de l'élément racine du schéma (i.e. l'intitulé du formulaire).

2. Transformer chaque élément ComplexType en nœud non terminal. Cette règle est appliquée de manière récursive à ses sous éléments.
3. Traduire tous les éléments simples ainsi que les attributs en nœuds terminaux.
4. Identifier le type de lien entre deux nœuds de l'arbre suivant la cardinalité du nœud fils. Si la cardinalité maximale est "maxoccurs=1", le lien est monovalué, si la cardinalité est "maxoccurs=n", le lien est multivalué.

5. Enrichissement de schéma relationnel

Le but de cette phase est d'augmenter la sémantique du schéma relationnel. Nous pouvons maintenant extraire la sémantique des données à partir des schémas de formulaires. Cette sémantique sera utilisée pour restructurer et enrichir sémantiquement le schéma relationnel de la base de données.

5.1 Extraction de la sémantique des formulaires.

Ce processus d'extraction permet de déterminer les relations et leurs clés respectives, ensuite d'extraire les dépendances fonctionnelles ainsi que les dépendances d'inclusion.

5.1.1 Identification des relations du formulaire

L'identification des relations d'un formulaire consiste à déterminer l'équivalence et/ou la similitude entre les nœuds de sa structure arborescente et les relations du schéma de la base de données (Malki, *et al.,* 2002). Un nœud est soit :

- équivalent à une relation de la base, i.e. que les deux entités (nœud et relation) ont le même ensemble d'attributs.
- similaire à une relation, i.e. que l'ensemble de ses attributs est une partie de celui de la relation.
- un ensemble de relations, i.e. que l'ensemble de ses attributs regroupe plusieurs relations.

Il se trouve que certaines relations du formulaire, ne vérifient pas ces règles de similarités avec les relations du schéma de la base de données. Ainsi, nous faisons intervenir le concepteur pour pouvoir extraire le reste des relations du formulaire. En appliquant ce processus sur la structure hiérarchique du formulaire "Booking-form" et le schéma physique de la base de données, nous obtenons le sous-schéma suivant :

Passenger (PassengerID, FirstName, LastName, Age)
Departure-City (CityID, DepartureCityName)
Arrival-City (CityID, ArrivalCityName)
Date (DepartureDate)

À partir du tableau "Program of flights", nous obtenons le sous-schéma suivant :

Departure-Hour (HourID, type)
Arrival-Hour (HourID, type)
Plane (PlaneID, Capacity)
Flight (FlightNumber, DepartureCityID, ArrivalCityID, Dep_HourID, Arr_HourID, PlaneID)

À partir de la relation entre les structures hiérarchiques des formulaires "Booking Form" et "program flight" nous obtenons le sous-schéma suivant :

Book (PassengerID, FlightNumber, DepartureDate, Class)
Leaving-From (FlightNumber, DepartureCityID)
Going-To (FlightNumber, ArrivalCityID)

5.1.2 Extraction des Dépendances Fonctionnelles

L'extraction des dépendances fonctionnelles (notées DFs) à partir de l'extension d'une base de données a fait l'objet de plusieurs travaux de recherche (Mannila, *et al.,* 1992, Petit, *et al.,* 1995). Dans notre approche nous utilisons l'algorithme présenté dans (Malki, *et al.,* 2002) pour réduire le coût de découverte des DFs en remplaçant les instances de la base de données par celles des formulaires dont la taille est nettement inférieure. En appliquant cet algorithme sur le schéma partiel du formulaire "Booking Form" ainsi que ses instances, on découvre la DF non triviale: PlaneID → FlightNumber, ce qui signifie qu'un avion assure seulement un vol.

5.1.3 Extraction des dépendances d'inclusion

Les dépendances d'inclusion (noté DIs) permettent de mettre en correspondance des valeurs stockées dans des relations différentes. Dans notre approche orientée formulaire, les attributs des DIs sont choisis à partir des clés primaire et étrangères des schémas partielles. Le coût de test des DIs est beaucoup plus optimisé par rapport aux autres approches (Mannila, *et al.,* 1992 ; Chiang *et al.,* 1994) car ce test se fait sur les instances des formulaires dont la taille est moins importante que celle de la base de données (Malki, *et al.,* 2002). En appliquant cet algorithme sur le sous-schéma du formulaire "Booking Form" ainsi que ses instances, on découvre l'ensemble des DIs :

DepartureCity.CityID << City.CityID
ArrivalCity.CityID << City.CityID

5.2 Enrichissement et restructuration du schéma relationnel

L'objectif de cette phase est de fournir un nouveau schéma relationnel restructuré et plus riche sémantiquement. Cette étape utilise la sémantique extraite dans la phase précédente pour :

- restructurer le schéma relationnel en le transformant en troisième forme normale à travers les dépendances fonctionnelles retrouvées.
- clarifier le schéma relationnel de la base de données en remplaçant les noms d'attributs et de relations par ceux plus explicites du schéma de formulaire. (e.g l'attribut "DC-Name" est remplacé par "DepartureCityName" qui est plus significatif).
- injecter les nouvelles contraintes extraites aux attributs des relations. Ces contraintes deviendront axiomes dans l'ontologie finale.
- ajouter les dépendances d'inclusion extraites au schéma relationnel. Ces dépendances révéleront par la suite des relations d'héritage dans l'ontologie.

Ainsi, le nouveau schéma relationnel enrichi devient : (Voir Table 2).

Table 2. *Schéma relationnel enrichi de la base de données*

Passenger (PassengerID, FirstName, LastName, Age) City (CityID) Departure-City (*CityID*, DepartureCityName) Arrival-City (*CityID*, ArrivalCityName) Date (DeparatueDate) Hour (HourID) Departure-Hour (*HourID*, type) Arrival-Hour (*HourID*, type) Company (CompagnyID, Address, Phone) Plane (PlaneID, *CompagnyID*, Capacity) Leaving-From (*FlightNumber, DepartureCityID*) Going-To (*FlightNumber, ArrivalCityID*) Flight (FlightNumber, *DepartureCityID, ArrivalCityID, DepartureHourID, ArrivalHourID, PlaneID*) Book (*PassengerID, FlightNumber, DepartureDate*, Class)	*Functional dependencies* PlaneID → FlightNumber ... *Inclusion dependencies* DepartureCity.CityID << City.CityID ArrivalCity.CityID << City.CityID Departure-Hour.HourID<< Hour.HourID Arrival-Hour.HourID << Hour.HourID ... *Constraints* Departure-City : NotNull. Arrival-City : NotNull. ...

6. Construction de l'ontologie OWL

6.1 *Modèle relationnel vs Ontologie*

Avant de présenter la méthode de construction de l'ontologie, nous rappelons de façon brève les notions du modèle relationnel en les comparant à ceux de l'ontologie. Dans le modèle relationnel (Codd, 1970), une base de données relationnelle se compose : d'un ensemble de relations R, d'un ensemble d'attributs AR, d'un ensemble de types de base TR, d'une fonction *attr* : R → AR×AR qui retourne les attributs de relations, d'une fonction *dom* : AR → TR qui retourne le type d'attributs, d'une fonction PK : R → AR×AR qui retourne les clés primaires de relations, et d'une fonction FK : R → AR×AR qui retourne les clés étrangères de relations. De plus il existe des relations de dépendance, entre les données des attributs dans le modèle relationnel.

Définition. Soit deux relations R_i et R_j dans la base de données, supposons que $A_i \subseteq attr(R_i)$ et $A_j \subseteq attr(R_j)$, $t_i(A_i)$ exprime les valeurs du tuple t_i pour les attributs A_i, et $t_j(A_j)$ exprime les valeurs du tuple t_j pour les attributs A_j. Si pour chaque $t_i(A_i)$ dans R_i, si il existe dans R_j $t_i(A_i) = t_j(A_j)$, alors il existe une dépendance d'inclusion entre A_i et A_j, notée $R_i(A_i) << R_j(A_j)$. Enfin des contraintes peuvent être exprimées sur les attributs du modèle relationnel (e.g. NOT NULL, UNIQUE, etc.)

D'autre part la structure d'une ontologie est un ensemble O = {C, R, Hc, *rel*, Ao} (Maedche, 2002). Où C est un ensemble fini de concepts ; R est un ensemble fini de relations; Hc désigne la hiérarchie ou la taxonomie de concepts, qui est une relation dirigée $Hc \subseteq C \times C$; *rel* définit des relations non taxonomiques entre les concepts. Ao est un ensemble d'axiomes, qui est exprimé dans un langage de logique du premier ordre. Basé sur cette structure, l'ontologie comporte un ensemble d'instances utilisées pour représenter l'extension des concepts. Au cours du processus de définition d'un langage Web standard pour les ontologies, un certain nombre de versions intermédiaires de langage ont été définies (OIL, DAML, DAML+OIL, etc.). Dans cet article nous adoptons la dernière norme recommandée par W3C, OWL (Ontology Web Language) (Smith, et al., 2003), contrairement à (Benslimane, et al., 2005a) où nous avons utilisé la logique de frame, comme langage de description d'ontologie.

6.2 *Règles de construction d'ontologie*

Dans ce qui suit nous présenterons un ensemble de règles de construction d'ontologie. Sans passer par un modèle intermédiaire, comme dans (Benslimane, *et al.,* 2005b), ces règles sont organisées en cinq groupes.

6.2.1 *Règle de construction de classes*

Règle 1. Une classe OWL C_i peut être créée à partir d'une relation R_i, si une des conditions suivantes est satisfaite :

(i) $|PK(R_i)| = 1$;

(ii) $|PK(R_i)| > 1$, et il existe A_i, telle que $A_i \in PK(R_i)$ et $A_i \notin FK(R_i)$.

6.2.2 *Règles de construction de propriétés*

OWL distingue deux types de propriétés : (owl:ObjectProperty) reliant des individus, et (owl :DatatypeProperty) reliant des individus à des types de données.

Règle 2. Soient R_i et R_i deux relations, si $R_i(A_i) \subseteq R_i(A_i)$ et $A_i \not\subset PK(R_i)$ sont satisfaites, alors une propriété d'objet P est créée à base de A_i. Supposons que les classes correspondantes à R_i et R_j sont C_i et C_i, alors ces dernières correspondent respectivement au domaine et au "range" (l'intervalle de valeurs) de P.

Règle 3. Soient R_i et R_j deux relations, deux propriétés d'objet peuvent être créées "has-part" et "is-part-of", si les deux conditions suivantes sont satisfaites :

(i) $|PK(R_i)|>1$;
(ii) $FK(R_i) \subset PK(R_i)$, $PK(R_j) \subseteq FK(R_i)$, où $FK(R_i)$ fait référence à R_j.

Supposons que les classes correspondantes à R_i et R_i sont respectivement C_i et C_j, le domaine et le "range" de "is-part-of" sont C_i et C_i et le domaine et le "range" de "has-part" sont C_j et C_i. "is-part-of " et "has-part" sont deux propriétés inverses.

Règle 4. Soient R_i, R_i et R_k trois relations, si $A_i = PK(R_i)$, $A_i = PK(R_i)$ et $A_i \cup A_i = FK(R_k)$, alors deux propriétés d'objet P_i' et P_i'' sont créées à base R_k. Supposons que les classes correspondantes à R_i et à R_i soient respectivement C_i et C_i, le domaine et le "range" de P_i' sont alors C_i et C_i, alors que le domaine et le "range" de P_j'' sont C_j et C_i. P_j' et P_j'' sont deux propriétés inverses.

Règle 5. Soient les relations R_1, R_2…, R_i et R_j, si $A_1 = PK(R_1)$, $A_2 = PK(R_2)$…, $A_i = PK(R_i)$ et $A_1 \cup A_2 \cup \ldots \cup A_i = FK(R_j)$, alors les propriétés d'objet P_1, P_2,…, P_i sont créées. Le domaine de chaque propriété d'objet P_i est la classe C_j correspondant à R_j et le range est la classe C_i correspondant à R_i.

Règle 6. Soit C_i une classe d'ontologie, $DP(C_i)$ l'ensemble des ses propriétés "datatype". Supposons que C_i correspond aux relations R_1, R_2…, R_i dans la base de données, alors chaque attribut dans R_1, R_2…, R_i, (non transformé en propriété d'objet par les règles 2 ou 5) sera transformé en propriété "datatype" de la classe C_i. Le domaine et le "range" de chaque propriété P_i sont respectivement C_i et $dom(A_i)$, où $P_i \in DP(C_i)$ et $A_i \in attr(R_i)$.

6.2.3 *Règle de construction des relations d'héritage*

Règle 7. Soient R_i et R_j deux relations, supposons que $P_i = PK(R_i)$ et $P_j = PK(R_j)$, si $R_i(P_i) << R_j(P_j)$ est satisfaite, alors la classe correspondante à R_i est une sous-classe de la classe correspondante à R_j.

6.2.4 *Règles de construction des axiomes*

Dans OWL, une propriété rattachée à une classe peut être contrainte par des restrictions de cardinalité au domaine exprimant le nombre d'instances minimum "minCardinality" et maximum "maxCardinality" qui peuvent participer à la relation.

Règle 8. Soit la relation R_i et l'attribut $A_i \in attr(R_i)$, si $A_i = PK(R_i)$ ou $A_i = FK(R_i)$, alors les cardinalités "minCardinality" et "maxCardinality" de la propriété P_i correspondant à A_i sont égales à 1.

Règle 9. Soit la relation R_i et l'attribut $A_i \in attr(R_i)$, si A_i est déclarée comme "NOT NULL", alors la cardinalité "minCardinality" de la propriété P_i correspondant à A_i est égale à 1.

Règle 10. Soit la relation R_i et l'attribut $A_i \in attr(R_i)$, si A_i est déclarée comme UNIQUE, alors la cardinalité "maxCardinality" de la propriété P_i correspondant à A_i est égale à 1.

6.2.5 *Règles de construction des instances*

L'objectif de cet ensemble de règles est la création des instances de l'ontologie. Le processus de migration de données s'effectue en deux phases par l'application successive des règles suivantes :

Règle 11 : A chaque instance un identifiant unique est affecté. Cette règle traduit les valeurs de tous les attributs, sauf ceux des clés étrangères.

Règle 12 : les relations entre les instances sont élaborées en utilisant les données des tuples des clés étrangères de la base de données. Ceci est réalisé en utilisant une fonction transformant ces clés en identifiants ontologiques.

7. Conclusion et travaux futurs

Dans cet article nous nous sommes intéressés au problème d'automatisation de la génération d'ontologies en appliquant des techniques de retro ingénierie pour construire des ontologies décrites en OWL à partir d'applications Web de données-intensives. Nous avons présenté les détails du processus semi-automatique de construction d'une ontologie OWL correspondant à une base de données relationnelle en analysant les formulaires HTML relatifs à l'application. Ce processus s'articule autour de trois étapes : i) extraction du schéma de formulaires, par l'analyse des pages HTML ; ii) enrichissement du schéma relationnel de la base de données à travers la sémantique du schéma de formulaires ; iii) construction de l'ontologie OWL en appliquant un ensemble de règles de transformation sur le schéma relationnel enrichi. L'objectif de cette démarche et de rendre exploitables par les machines, les bases de données relationnelles disponibles sur le Web, réduisant ainsi le coût de construction des ontologies.

Cependant, la qualité de la structure de l'ontologie obtenue dépend de la qualité des données de la base utilisée. Une validation de la sémantique de l'ontologie acquise est nécessaire. Ainsi un raffinage de la structure ontologique obtenue est nécessaire. En perspectives nous suggérons l'utilisation des entrepôts de connaissances lexicales pour valider la sémantique de l'ontologie obtenue (e.g. WordNet).

8. REFERENCES

Astrova, I. Reverse Engineering of Relational Databases to Ontologies , *In: proceeding of the 1st European Semantic Web Symposium* (ESWS), Heraklion, Crete, Greece, LNCS, 2004, 327–341.

Astrova, I., Stantic, B. An HTML Forms driven Approach to Reverse Engineering of Relational Databases to Ontologies, *In: proceeding of the 23rd IASTED International Conference on Databases and Applications* (DBA), eds. M. H. Hamza, Innsbruck, Austria, 2005, pp. 246- 251

Benslimane, S.M., Malki, M., Amar Bensaber, D. Automated Migration of Data-Intensive Web Pages into Ontology-Based Semantic Web: A Reverse Engineering Approach. *In, Meersman R., Tari Z. et al.,(eds.)*,ODBASE, vol. 2, LNCS 3761, pp. 1640 - 1649, 2005. Springer Verlag.

Benslimane, S.M., Malki, M., Rahmouni, M.K. Benslimane, D. Building domain-specific ontology from data-intensive Web site: An HTML forms-based reverse engineering approach, *In: Proceedings of the International Conference on Signal-Image Technology & Internet- Based Systems* (SITIS'05/IEEE), Yaoundé, Cameroon, 2005.

Biskup, J. Achievements of relational database schema design theory revisited. *Semantics in Database, Springer Verlag*, 1998.

Chiang, R.H.L., Barron, T.M., Story, V.C. Reverse engineering of relational databases: extraction of an EER model from a relational database. *Data and Knowledge Engineering,* 1994.

Codd, EF. A relational model of data for large shared data banks". *CACM* 13 No.6, 1970.

Embley, D. Toward Semantic Understanding – An Approach Based on Information Extraction, *In: Proceedings of the 15th Australasian Database Conferenc*e, 2004, 3–12.

Florescu, D., Levy, A., Mendelzon, A. Database Techniques for the World Wide Web: A Survey, *ACM SIGMOD Record*, Vol. 27, No. 3 (1998) 59–74

Hainaut, J. Henrard, J., Hick, J.M., Roland, D., Englebert, V. Database Design Recovery, *In: Proceedings of the 8th Conference on Advanced Information Systems Engineering* (CAiSE), Heraklion, Crete, Greece, LNCS, 1080, 1996, 272–300.

Haustein, S. and Pleumann, J. Is participation in the Semantic Web too difficult? *In First International Semantic Web Conference*, number 2342 in LNCS, pages 448-453, Sardiana, Italy, June 2002. Springer-Verlog.

Kashyap, V. Design and Creation of Ontologies for Environmental Information Retrieval, *Proc. 12th Workshop on Knowledge Acquisition, Modeling and Management* (KAW), Banff, Alberta, Canada, 1999.

Maedche, A. *Ontology learning for the Semantic Web*. Boston: Kluwer Academic Publishers. 2002

Malki, M., Flory, A., Rahmouni, M.K. Extraction of Object-oriented Schemas from Existing Relational Databases: a Form-driven Approach, *INFORMATICA, International Journal (Lithuanian Academy of Sciences)* pp 47-72, Vol. 13(1), 2002.

Mannila, H., Räihä, K.J. *The Design of Relational Databases*. Addison-Wesley publishing, England, 1992, 318 pages.

Mfourga, N. Extracting entity-relationship schemas from relational databases: a form-driven approach. *In Proc. of Working Conf. on Reverse Engineering* WCRE'97 1997.

Michalski, R. *A Theory and Methodology of Inductive Learning. Machine Learning,* vol.1, Eds. J.G.Carbonel, R.S.Michalski and T.M.Mitchel, Palo Alto (1983) 83-134,.

Muller, R. *Database Design for Smarties: Using UML for Data Modeling*, Morgan Kaufmann (1999).

Petit, J.M. Toumani, F. Kouloumdjian, J. Relational database reverse engineering: a method based on Query analysis. *International Journal of Cooperative Information System*, 4(2,3), 287–316, 1995.

Premerlani, W. and Blaha, M. An Approach for Reverse Engineering of Relational Databases, *In: Communications of the ACM,* Vol. 37. No. 5 (1994) 42–49

Rubin, D.L., Hewett, M., Oliver, D.E., Klein, T.E, Altman, R.B. Automatic data acquisition into ontologies from pharmacogenetics relational data sources using declarative object definitions and XML. *In: Proceedings of the Pacific Symposium on Biology*, Lihue, HI, Eds. R.B. et al l (2002).

Smith, M.K., Welty, C., McGuinness, D.H Eds. *OWL Web Ontology Language Guide*. W3C Proposed Recommendation, December 2003.

Stojanovic, L., Stojanovic, N., Volz, R. Migrating Data-intensive Web Sites into the Semantic Web, *Proc. 17th ACM Symposium on Applied Computing*, Madrid, 2002.

Tijerino, Y.A. et al., *Towards Ontology Generation from tables*. Springer Science+Business Media B.V, Kluwer Academic publishers, September 2005, 261 - 285.

Volz, R., Handschuh, S., Staab, S., Stojanovic, L., Stojanovic. N. Unveiling the hidden bride: deep annotation for mapping and migrating legacy data to the Semantic Web, *Journal of Web Semantics: science, services and agents on the Word Wide Web 1* (2004) 187-206.

Wang, J., Lochovsky, F.: Data Extraction and Label Assignment for Web Databases, In: *Proceedings of the 12th International Conference on World Wide Web* (WWW), Budapest, Hungary, 2003, 187–196.

Yang, Y., Zhang, H. HTML Page Analysis Based on Visual Cues, *In:Proceedings of the 6th International Conference on Document Analysis & Recognition* (ICDAR), Seattle, WA, USA, 2001, 859–864.