# Data Integration Targeting a Drug Related Knowledge Base

Olivier Curé and Raphaël Squelbut

Université de Marne-la-Vallée, Laboratoire ISIS
5, boulevard Descartes
Marne-la-Vallée 77454 France
Email: (ocure,rsquelbu)@univ-mlv.fr 77454 Champs-sur-Marne, France

**Abstract.** We present a patient-oriented computer-based medical system which proposes advices on mild clinical signs treatment and medication. Therefore, this system can be considered a self-medication assistant tool. In a nutshell, this web application validates drug consumptions of a given patient, based on patient information stored in an electronic health care record, with a drug and symptom knowledge base. The efficiency and accuracy of the knowledge base inferences depend on the quality, quantity and recency of the drug instances. A practical source for these information are databases. Thus we developed a data integration solution which enables the mapping of relational databases to a Description Logics knowledge base.

## 1 Introduction

Nowadays, it becomes obvious that the french public health system has to evolve in order to survive. Among possible evolutions, the act to provide assistance and reliable services to the general public is promising. This approach may result in giving more responsabilities to the patient and hence reducing the costs of care for the health system. This issue is of central importance in countries where drug over-consumption is a general practice. In France, this situation is partly due to a high rate of drug reimbursements by the social security system. Consequently, the adoption by the general public of such services may reduce costs for both the patient and the public health system, prevent medical errors due to drug interactions, and improve efficiency and quality of patient treatment. We believe that these facts can be generalized to most industrial countries and hence will soon cover a central aspect of computer-based medical systems.

We have developed, with the clinical pharmacology department at the Cochin hospital in Paris (France), a system named IMSA (Interactive Multimedia System for Auto-medication) [3] and its extension XIMSA (eXtended IMSA) [4] enabled with Semantic Web technologies. Both of these systems belong to this category of medical information systems. The latest version of this system embeds a drug consumption checking service. The purpose of this module is to check, according to data stored for a given patient, the adequacy, in terms of drug characteristics, of a medication (self) prescription. This service requires inferences on updated and accurate drug knowledge to detect clinical and drug contraindications, side effects, etc..

We have decided to store these data in a knowledge-based system to enable reasoning from explicit as well as implicit data. As databases become widely used, there is a need to translate data to the knowledge base and therefore to embed a data integration solution in the framework. This solution ensures that the right information is available at the right time from the right source of information. Consequently an ontology has to be designed from the data sources via a mapping solution, permitting tuples from these sources to populate the knowledge base.

The representation of this ontology is based on Description Logics (DL) [1], a fragment of first order logic, which are advocated as the key technology for realizing the Semantic Web. Standardization efforts within the World Wide Web Consortium (W3C) have resulted in the Web Ontology Language (OWL). Numerous tools, such as editors and reasoners, are already available for this language and thus Semantic Web applications can realistically be implemented with recognized standards.

This paper is organized as follows. In section 2, we present and justify the drug consumption checker tool within the context of the XIMSA system. In section 3, we present our data integration system which ensures that data contained in relational databases are integrated in XIMSA's ontology. In section 4, we present some general integration issues related to our information migration solution. Finally, we conclude this paper with some perspective on future works.

## 2 Drug consumption checker application

The Drug Consumption Checker Application, henceforth DCCA, is a service proposed within XIMSA, and is aimed at the general public. This system assists but is not limited to a popular activity in most industrial countries : self-medication. In this paper, we consider self-medication as the health activities to treat oneself with or without drugs. Within this system, the goal of the DCCA module is to provide patients with a tool that controls the adequacy of a drug (self) prescription. The architecture of the DCCA service is supported by a drug ontology and a Simplified Electronic Health Record (SEHR), especially designed for the XIMSA system.

The goal of the SEHR is to store health related information concerning a particular patient. The formalism adopted for the SEHR is the semi-structured XML language. An instance of such a document stores three different categories of patient information :

- general information concerning the owner of this document : social security number, first and last name, gender, date of birth, etc..
- medical information concerning known diseases, allergies, current states (pregnancy, breast feeding, etc.).
- drug consumption information which distinguishes discrete and continuous (life long treatments) consumptions. Both consumptions require the start

date and dosage of the treatment. For a discrete consumption, additional fields concerning the treatment duration and prescription source (either the patient or a health care professional) are required. We consider that non-discrete treatments are the responsability of health care professionals.

Example 1 proposes an XML extract from a SEHR which highlights the consumption of the *Marsilid*© drug between november 1st and november 10th, thus it is a discrete prescription, with a dosage of one pill per day. This drug is identified by the french identifier for drug products (cip code) 3442856 and has been prescribed by a general practitioner.

*Example 1.* SEHR discrete consumption example

```
<discretePrescription >
<prescription >
<idnum >3442856 </idnum >
<nomMed >Marsilid </nomMed >
<posologie >1 comprime par jour </posologie >
<prescriptionSource >Dr XXX </prescriptionSource >
<datedebut >1/11/2005 </datedebut >
<datefin >10/11/2005 </datefin >
</prescription >
...
</discretePrescription >
```

The drug and symptom knowledge base is central to the architecture of XIMSA as it is being used to make inferences in various services. In this paper, we concentrate solely on the DCCA service and on the drug knowledge base. For the following scenario, we consider that our knowledge base contains the Summary of Product Characteristics (SPC) of all french drugs :

*Example 2.* On november 7th, 2005, a patient connected to the DCCA service wants to self-prescribe the *Pulmodexane*© drug. He selects this drug from the system's graphical user interface, accepts the default dosage and validates his choice. The system then checks the patient's SEHR. We consider that this document contains no continuous prescription and that the only discrete prescription entry corresponds to the one in example 1. The inference engine advises not to use this drug at the moment because a contraindicated drug is currently being medicated. The result of this inference is based on the analysis of the characteristics of the Recommended International Non-proprietary Names (RINN) of *Pulmodexane*, i.e. *dextromethrophan* and the *Marsilid*, i.e. *iproniazide.* Navigating the knowledge base, the system is able to find that these two chemical substances are contraindicated one another. Additionally, the system proposes some drugs belonging to the *Pulmodexane*'s therapeutic class, i.e. anti-coughing, which are coherent with a current *marsilid* treatment.

The quality of the DCCA inferences results from the sufficiency, recency and accuracy of the data contained in the knowledge base. To ensure these quality requirements, it is necessary to consider the knowledge base's domain : pharmacology and its market. In France, the drug market is rapidly evolving due to frequent addition, modification and deletion of drugs, drug switches, reimbursement rate changes, emergence of the generics market, etc.. Given these characteristics, the most reliable source of information for the population of this knowledge base are drug related databases. In the context of XIMSA, we already maintain a drug database, named *self*, which stores all the information contained in the SPCs (posology, composition, contraindication, side effects, therapeutic class, price, social security system reimbursement rate, pharmaceutical laboratory, etc.) plus some extra information provided by collaborating health care professionals, i.e. comments on drugs as well as drug rating. Although databases are widely used in the drug industry, there does not exist an exhaustive and up-to-date french drug database, even from french administrations. Hence, in order to fulfill DCCA's recency and exhaustivity requirements, it is pertinent to translate and integrate data contained in multiple drug databases in XIMSA's knowledge base.

## 3 Data integration

The development of the XIMSA application motivated the development of the DBOM (DataBase Ontology Mapping) system. DBOM is a fully-implemented and domain-independent application which enables the creation, population and maintenance of a DL-based ontology from database sources. This solution is based on a declarative mapping document. In order to present our data integration solution, it is necessary to provide some formal and general aspects about the components involved in the mapping.

### 3.1 Preliminaries

The prototype we have developed adopts a database reverse engineering approach which enables to design an ontology from a set of Entity-Relationship diagrams. This design is materialized through a mapping file whose purpose is to build a DL ontology schema and populate it from tuples of source databases . We now characterize the system underlying the DBOM system and its cornerstone : the mapping language.

**Definition 1.** *The DBOM system is supported by a migration system $\mathcal{MI}$ which is a triple $(\mathcal{S}, \mathcal{O}, \mathcal{M})$, where :*

- *$\mathcal{S}$ is a set of source schemas of relational databases.*
- *$\mathcal{O}$ is the (target) ontology schema formalized in OWL DL.*
- *$\mathcal{M}$ is a set of formulas of a language $\mathcal{L}_{\mathcal{M}}$ over $\mathcal{S}$ and $\mathcal{O}$.*

The definition of $\mathcal{MI}$ emphasizes relations with data exchange [12] and data integration [13] systems. We now contrast the DBOM approach with the comparison of data exchange and integration provided in [8] :

- as in both data exchange and integration, the source schemas are given and the mapping is a set of formulas constructed by a human expert.

– as in data integration, the ontology (target) schema is a reconciliation of the sources and is constructed from the processing of the source schemas given a mapping.
– as in data exchange, the target instances are materialized, while they are virtualized in the case of data integration.

The target schema is an ontology designed in a DL. This family of knowledge representation formalisms allow the represention and reasoning over domain knowledge in a formally and well-understood way. In the DBOM approach, a DL ABox (assertional box or extension of the knowledge base) is considered as a view of the relational database. Our contribution to this issue lies in the possibility to richly axiomatized the terminology; thus permitting the creation of expressive ontologies, corresponding to the $\mathcal{SHOIN}$(D) DL, equivalent to the OWL DL language. Another interesting feature is the solution proposed to maintain the synchronization between the database tuples and the ABox of the knowledge base [5]. We assume readers are familiar with the semantics of DL, though we recall that the syntax for concepts in $\mathcal{SHOIN}$(D) [11] are defined as follows, where C$_i$ is a concept, A is an atomic concept, R$_i$ is an object role, S is a simple object role, T is a datatype role, D is a datatype, o$_i$ is an individual and n is a non-negative integer :
$C \rightarrow$ A | $\neg\ C_1$ | $C_1 \sqcap C_2$ | $C_1 \sqcup C_2$ | $\exists\ R_1$.C | $\forall\ R_1$.C | $\geq$ n S | $\leq$ n S | $\{o_1, .., o_m\}$ | $\geq$ n T | $\leq$ n T | $\exists$ T.D | $\forall$ T.D

The reason of our interest in the $\mathcal{SHOIN}$(D) DL is its syntactical equivalence with the OWL DL language [6], an expressive ontology language developed by the W3C and which is already supported by numerous tools (editors, reasoners, etc.). The choice of this formalism is motivated by our need to reason over web compliant data represented in an expressive, formal and decidable knowledge representation language. It is important to emphasize that semantic integration plays a key role in the growth of the Semantic Web and thus motivates many researches in this field, [5] proposes a study of relevant projects in this field.

Finally, regarding the source schema, we assume we have a fixed database schema P=$\{P_1, .., P_n\}$ where $P_i$, with $1 \leq i \leq n$, are predicates corresponding to the database relations. We also have a fixed, possibly infinite database domain D and a fixed set of built-in predicates B $(=, \neq, <, >, \leq, \geq)$. We define the first-order language $\mathcal{L}$ as $P \cup D \cup B$. Each predicate of $\mathcal{L}$ has an arity, i.e., the number of arguments taken.

### 3.2 Integration of the *self* database

The idea of the DBOM solution is to integrate data contained in relational databases in an ontology as expressive as the $\mathcal{SHOIN}(D)$ DL.

Concretly, data contained in tuples of P, a database instance, are mapped to concepts $C_i$ and roles $R_i$ of the ontology ([1] chap.16). The population of the ABox is performed via the execution of queries contained in the definition of concrete concepts and (binary) roles. The extract of the *self* database, presented in example 3, focuses on drugs and their chemical substances. Primary keys of relations are underlined and they correspond to a french drug identifier code (cip) and an international code for the identification of chemical substances (atcCode). The Anatomical Therapeutic Chemical (ATC) system [14] proposes an international classification of drugs and is part of WHO's initiatives to achieve universal access to needed drugs and rational use of drugs. In this classification, drugs are classified in groups at five different levels. The subset P' of the relational schema P of the *self* database proposes the last level which corresponds to chemical substances. Finally, the *drugRate* attribute contains a drug grade based on a tolerance/efficiency ratio and is designed by health care professionals of our research group.

*Example 3.* Schema extract of the *self* database
drug ($\underline{\text{cip}}$, drugName, drugPrice, drugRate)
rinn ($\underline{\text{atcCode}}$, substanceName)
rinnToDrug ($\underline{\text{cip}}$, $\underline{\text{atcCode}}$)

Given the relational schema of example 3, we propose a possible mapping in example 4. This mapping is presented in the form of conjunctive queries as they are more readable and concise than the XML serialization. The mapping provides a complete freedom to the ontology designer and offers the ability to integrate some or all $P'_i$ of P'. It is also possible to define concepts and properties that are not mapped to any $P'_i$ of P', meaning that these members must be abstract because no instantiations will be possible, as there are no queries attached. The abstract (and concrete) members feature have the same meaning than in object-oriented programming, thus abstract member can not be instantiated while concrete ones can. These definitions enable to design rich generalization/specialization relations between members. Example 4 only proposes concrete members as we primarilly focus in this paper on data integration and not on knowledge base inferences. For readability reasons, concepts start with an uppercase letter and properties are lowercased.

*Example 4.* A valid DBOM mapping file for example 3's relational schema
*Drug* $\rightsquigarrow$ { (W,X,Y,Z) | drug(W,X,Y,Z)}
*Rinn* $\rightsquigarrow$ { (X,Y) | rinn(X,Y)}
*rinnToDrug* $\rightsquigarrow$ { (X,Y) | rinnToDrug(X,Y)}

This example emphasizes that the integration solution adopts a Global-As-View (GAV) approach. The GAV solution means that each assertion in $\mathcal{M}$ relates an element of the target schema $\mathcal{O}$ to a query over a source schema $\mathcal{S}$. The counterpart of GAV is the Local-As-View approach (LAV) where the mapping specifies the content of the source in terms of the global schema. A comparison of the two approaches, and possible translations from one to the other, is available in [2].

The XML serialization of example 4's *Drug* concept is presented in example 5. In this example, we consider that the prolog of the mapping has defined the following elements :

- the *hasName*, *hasPrice*, *hasGrade* datatype properties, binary properties where the domain is an ontology concept instance and the range is an XML schema data type corresponding to the domain D of the $P_i$ mapped attribute.
- the information (database driver, hostname, login, password) necessary for a connection to the *self* database. An alias, named *selfDB*, is declared for this connection.

*Example 5.* XML serialization of the *Drug* concept

```
<class namespace=”drug” className=”Drug” >
<instance dbSrc=”selfDB” query=”SELECT * FROM drug;” >
<id >
<field value=”1”/ >
</id >
<data >
<field value=”2” datatypeProperty=”hasName”/ >
<field value=”3” datatypeProperty=”hasPrice”/ >
<field value=”4” datatypeProperty=”hasGrade”/ >
</data >
</instance >
</class >
```

DBOM processes such an XML mapping in the following manner :

- given the attributes of the *instance* element (second line), a SQL query is performed for an identified database connection.
- the tuples resulting from this query processing are mapped to data type properties. A special data type property serves as a primary key, to relate unambiguously database tuples and knowledge base instances, an operation required by the maintenance solution. Agregated primary keys can be defined in DBOM mapping files.

The identification of the database connection for each SQL view definition enables to integrate several databases in a single mapping instance. This can be done by declaring several database connection aliases in the prolog.

### 3.3 Integrating several databases with DBOM

We now consider practical and concrete integrations in DCCA with the *self* database. This database does not contain all drugs available on the french market and may also need to integrate emerging standards. So it may be necessary to integrate data from additional databases. We can distinguish two integration situations :

- “instance integration” where the new source populates a given target schema with instances that are not yet available in the knowledge base.
- “schema integration” where the new source requires modifications of the target schema by adding concepts and properties. These newly created concepts and properties are then populated from this source.

An example of "instance integration" is proposed in the following scenario : a database containing some drugs, and their chemical substances, missing from the *self* database, has been found and is thus integrated. The schema of the database (*db1*) is proposed in example 6 :

*Example 6.* Schema of the *db1* database
drug (cip, drugName, drugPrice)
molecule (atcCode, moleculeName)
moleculeToDrug (atcCode, cip)

We emphasize that the schema of *db1* does not contain a drug grade column as it is a specificity of the *self* database. Given this relational schema, a possible mapping instance can integrate anti-coughing drugs containing the *dextromethorphan* chemical substance which can be unambiguously identified with the R05DA09 ATC code. This code extract uses a special representation of conjunctive queries where database names are prefixing attribute names.

*Example 7.* A valid DBOM mapping integrating the *self* and *db1* databases
*Drug* $\rightsquigarrow$ { (W,X,Y,Z) | SELF.drug(W,X,Y,Z)}
*Drug* $\rightsquigarrow$ { (X,Y,Z) | DB1.drug(X,Y,Z) $\wedge$ DB1.moleculeToDrug(W,X)
$\wedge$ W='R05AD09'}
*RINN* $\rightsquigarrow$ { (X,Y) | SELF.rinn(X,Y)}
*RINN* $\rightsquigarrow$ { (X,Y) | DB1.molecule(X,Y) $\wedge$ X='R05AD09'}
*drugToRinn* $\rightsquigarrow$ { (X,Y) | SELF.rinnToDrug(X,Y)}
*drugToRinn* $\rightsquigarrow$ { (X,Y) | DB1.moleculeToDrug(X,Y) $\wedge$ X='R05AD09' }

This mapping highlights several aspects of the integration with DBOM :

- the use of queries for the population of the knowledge base enables the design of 'filters', e.g. we only select drugs with the *dextromethorphan* chemical substance from *db1.*
- the use of set operations on the results of queries. The semantics of this mapping emphasizes the use of the union set operations which allows to combine the results of the several views for the same member, e.g. two definitions for the *Drug* concept.
- the drugs populated from the *db1* database do not contain grade values. In such a situation, the DBOM's processing does not assign any value to this drug, thus such a drug does not have a *hasGrade* data property. This is due to the fact that the first *Drug* definition has four attributes in its view while the second one only has three attributes.

A situation of "schema integration" corresponds to the need to integrate valuable data from health care institutions. We take as an example the integration of a DDD database. The Defined Daily Dose "is the assumed average maintenance dose per day for a drug used for its main indication in adults" [14]. The integration of this classification may serve to propose posology in the absence of such data for a particular drug. Although, the DDD has some drawbacks in the context of compound drugs ( with several RINNs), it still is an interesting candidate for integration. The relational schema of example 8 highlights the relation between the ATC/DDD system where each molecule is identified by an ATC code and characterized by the defined daily dose, unit of dosage and notes. Anotehr relation contains all the administration routes avalailable, e.g. oral.

*Example 8.* Schema of the *ddd* database
ddd (atcCode, ddd, unit, adminId, notes)
administrationRoute(adminId, adminRouteName)

We can propose a mapping instance for the *self* and *ddd* databases which assumes that :

- DDD information (dosage, unit, administration route and notes) are stored in a *Ddd* concept.
- the *adminRouteName* for an *adminId* is stored directly, in the form of a property, in the *Ddd* concept.
- we consider that all RINN that match a DDD are in the *self* database; otherwise the *ddd* database can easily be filtered before the mapping processing to match all the ATC code of the *self* database.
- a *rinnToDrug* property relates an *Rinn* instance to a *Ddd* instance.

*Example 9.* A valid DBOM mapping file for the *self* and *ddd* schemas
*Drug* $\rightsquigarrow$ { (W,X,Y,Z) | SELF.drug(W,X,Y,Z)}
*Rinn* $\rightsquigarrow$ { (X,Y) | SELF.rinn(X,Y)}
*Ddd* $\rightsquigarrow$ { (T,U,V,Z,X) | DDD.ddd(T,U,V,Y,X) $\wedge$ DDD.administrationRoute(Y,Z)}
*drugToRinn* $\rightsquigarrow$ { (X,Y) | SELF.rinnToDrug(X,Y)}
*rinnToDDD* $\rightsquigarrow$ { (X,X) | DDD.ddd(X,Y)}

A more efficient mapping proposition would merge all the information contained in *Ddd* instances in the corresponding *Rinn* instances. But this operation requires to write views in the GAV mapping joining relations from both databases. Such an operation is not yet implemented in the DBOM framework but is on our future works list.

Figure 1 proposes a graph extract of the knowledge base resulting from the processing of example 9's mapping. In the context of DCCA, graph navigation enables to make inferences using explicit as well as implicit knowledge. Enriching Figure 1's graph with a *contraindicatedWith* symmetric property between RINNs R05DA09 and N05AF06, we would be able to detect that any drug containing the *dextromethorphan* chemical substance is contraindicated with a drug containing the *iproniazide* substance.

## 4 Data integration issues in DBOM

DBOM faces all the commonly encountered problems of data integration which are based on heterogeneity, redundancy and inconsistency :

- structural heterogeneity is due to structure differences between sources, e.g. naming conflicts, data type conflicts, integrity conflicts, etc..

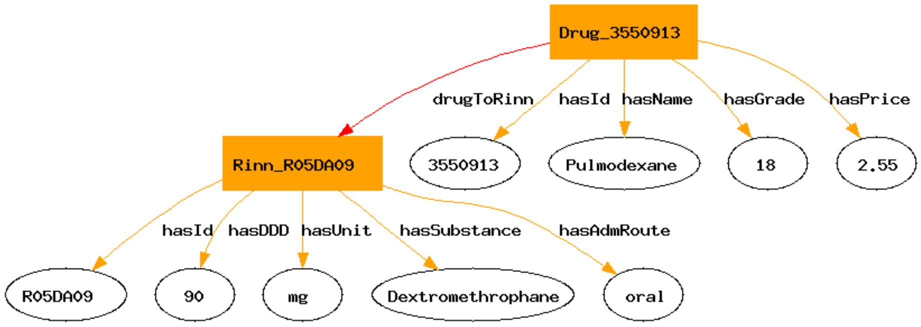

**Fig. 1.** Ontologie graph resulting from example 9's mapping

- semantic heterogeneity is due to different interpretations of the data and domain of sources, e.g. representation conflicts, subsumption conflicts, etc..
- redundancy is due the absence of common key identifier between sources for equivalent relations.
- inconsistency is due to the presence of different values for equivalent data in different source, incomplete information, etc..

Solving these heterogeneity problems is a complex and time-consuming task that is usually left to the user. In the XIMSA system, a particular attention is given to the selection of source candidates. They are generally proposed by health care professionals and validated by the user responsible for the mapping. Several operations, such as data cleaning [9] may be required to process the mapping. Whether these problems are solved manually or in a (semi-)automatic way, semantic is the key issue. This is a challenging problem because there are few reliable and non-subjective information sources for semantic :

- the creator of the database who are generally not accessible or have forgotten about the data.
- documentations are generally missing or tend to be outdated, sketchy and incorrect.

Many solutions are designed to solve these problems in a semi-automatic way, meaning that the user is involved in the processing of the system [7] in terms of validation of the results, providing some clues to solve problems, etc..

An interesting service proposed within DBOM is the maintenance of both the database sources and the knowledge base. This maintenance solution is described in [5] for a single database source. The idea of this maintenance is two fold :

- to automatically maintain the synchronization between the knowledge base instances and the database tuples. This means that whenever a tuple is updated (insertion, deletion and modification) in a source, the corresponding knowledge base instance has to be modified. [5] emphasizes that the synchronization can be delayed because of integrity constraints expressed on sources relational schema.

- to semi-automatically maintain the database tuples from detection of inconsistencies on the knowledge base's side. The semi-automatic aspect of this maintenance means that the intervention of a user is required to solve the inconsistency.

A practical scenario for the second form of maintenance is now proposed : "A health care professional collaborating to XIMSA inserts a new drug in the *self* database. This drug belongs to the 'anti-coughing' therapeutic class and contains the *iproniazide* chemical substance. This tuple is automatically translated in the form of a concept instance, and its properties, in the knowledge base. XIMSA's inference engine then tries to detect possible inconsistencies in the knowledge base and finds that the relation between the therapeutic class and the RINN was not unknown.". The treament of such inconsistencies is not automatized and XIMSA awaits from the user to correct or validate this new entry.

In a data integration context, the maintenance solution needs to efficiently identify the source of a knowledge base instance. This identification serves to correct the data at the source level, for example the modification of a newly inserted drug in the *self* database. The purpose of these updates is to maintain the *self* database consistent because it is exploited in several other situations : additional services in XIMSA, support of dynamic web sites, a book [10], etc.. These interventions can be processed only on updatable sources, as we are not granted to modify all integrated sources, e.g. a *ddd* database.

## 5 Conclusion

This paper present an extension to the DBOM system, formerly a migration tool from a unique database to a Semantic Web compliant (OWL DL) knowledge base. This extension now proposes to integrate several sources in an ontology and processes its instantiation from tuples of the sources. This approach considers the DL Abox as a relational view over the relational databases.

We have presented conceptual and functional aspects of this system via practical examples in the context of a drug prescription checking tool. Through data integration, the DCCA is ensured to access an up-to-date drug market knowledge base and hence supports efficient and high-quality patient reasoning procedures. This approach turned out to be very efficient in the context of XIMSA. As a self-medication tool, XIMSA aims to be used by the general public. Hence the content of the drug characteristics (SPCs) have been transformed and translated to be more easily understood by patients.

We are now thinking about developing a XIMSA version tailored to health care professionals. This tool would assist general practitioners to prescribe drugs to their patients in a more secure and cost-effective way. Some tests conducted with medicine students have shown that such a tool may be interesting to this community. We believe that DBOM's functionalities may help to integrate new drug concepts and standards from the pharmaceutical industry , e.g. the french drug efficiency rating (SMR) and amelioration (AMSR).

The DBOM system is also evolving on several aspects and we are currently :

– developing a Protégé plug-in to enable the design of mapping files with a graphical user interface. This plug-in would benefit from a collaboration with other OWL components (class, property tabs, visualization and rule solutions, etc.).
– studying the maintenance of the knowledge given some particular source schema modifications. This is a novel approach as at the moment our maintenance solution only tackles instance modifications.

## References

1. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. : *The Description Logic Handbook: Theory, Implementation and Applications* Cambridge University Press (2003).
2. Cali, A., Calvanese, D., De Giacomo, G.,Lenzerini, M. : *On the expressive power of data integration* Proceedings of ER 2002. pp 338-350.
3. Curé, O. : *Overview of the IMSA project, a patient-oriented information system* Data Science Journal 1(2): pp 66-75 (2002)
4. Curé, O. : *XIMSA : eXtended Interactive Multimedia System for Auto-medication* IEEE Computer-Based Medical System Symposium (CBMS) 2004. pp 570-575.
5. Curé, O., Squelbut, R. : *A database trigger strategy to maintain knowledge bases developed via dat a migration* EPIA 2005. LNAI 3808 pp 206-217.
6. Dean, M., Schreiber, G. : *OWL Web Ontology Language Reference*. 2004. W3C Recommendation 10 February 2004
7. Doan, A., Halevy, A. : *Semantic integration Research in the database community : a brief survey* AI Magazine, Special Issue on Semantic Integration, Spring 2005. pp 83-94.
8. Fagin, R., Kolaitis, P., Miller, R., Popa, L. : *Data exchange : semantics and query answering* Proceedings of ICDT 2003. LNCS 2572. pp 207-224.
9. Galhardas, H. Florescu, D., Shasha, D., Simon, E., Saita, C-A : *Declarative Data Cleaning: Language, Models, and Algorithms*. In Very Large Data Bases 2001. pp 371-380.
10. Giroud, J.P., Hagege, C. : *Le guide Giroud-Hagege de tous les médicaments* Editions du Rocher, Paris, France. 2001.
11. Horrocks, I., Sattler, U. : *A Tableaux Decision Procedure for SHOIQ*. Proceedings of IJCAI 2005. pp 448-453.
12. Kolaitis, P. : *Schema mappings, data exchange, and metadata management* In ACM Symposium on Principles of Database Systems (PODS) 2005. pp 61-75.
13. Lenzerini, M. : *Data integration : a theoretical perspective*. In ACM Symposium on Principles of Database Systems (PODS) 2002. pp 233-246.
14. WHO Collaborating Centre for Drug Statistics Methodology URL of Web site : http://www.whocc.no/atcddd/