Semi-automatic Generation of a Patient Preoperative Knowledge-Base from a Legacy Clinical Database

Matt-Mouley Bouamrane[1,2], Alan Rector[1], and Martin Hurrell[2]

[1] School of Computer Science
Manchester University, UK
{mBouamrane,Rector}@cs.man.ac.uk
[2] CIS Informatics, Glasgow, UK
martin.hurrell@informatics.co.uk

Abstract. We discuss our practical experience of automating the process of migrating a clinical database with a weak underlying information model towards a high level representation of a patient medical history information in the Web Ontology Language (OWL). The purpose of this migration is to enable sophisticated clinical decision support functionalities based on semantic-web technologies, i.e. reasoning on a clinical ontology. We discuss the research and practical motivation behind this process, including improved interoperability and additional classification functionalities. We propose a methodology to optimise the efficiency of this process and provide practical implementation examples.

1 Introduction

In the last two decades, in order to improve efficiency, cost-effectiveness and patient safety, health information and management systems (HIMS) and clinical decision support systems (CDSS) have steadily been moving towards greater standardisation and interoperability of clinical information. In recent years, in response to a combination of economic necessities and international health agencies policies, this overall process has been accelerating. There has been considerable progress towards the standardisation of information interchange formats with Health Level 7 (HL7), information structure (Clinical Document Architecture: CDA and Electronic Health Record: EHR) and information modelling and representation through the use of standard taxonomies and clinical terminologies (e.g. Snomed-CT[1]). However, many information systems still fail to have any significant impact in practice. System designers often underestimate the constraints imposed by existing work practices and legacy systems in use in the health services. In this paper, we discuss our practical experience of automating the process of migrating a clinical database with a weak underlying information model towards a high level representation of clinical information in the Web

[1] http://www.nlm.nih.gov/research/umls/Snomed/snomed_main.html

R. Meersman, T. Dillon, P. Herrero (Eds.): OTM 2009, Part II, LNCS 5871, pp. 1224–1237, 2009.

Ontology Language (OWL). The purpose of this migration is to enable sophisticated decision support functionalities based on semantic-web technologies, i.e. reasoning on a clinical ontology.

We have reported elsewhere our work on combining a preventive care software system for preoperative risk assessment of patient with a decision support ontology developed with a logic based knowledge representation formalism [1,2,3,4] We here describe our experience with the practical challenges encountered while integrating semantic web technology with a legacy clinical database. The paper is structured as follows: a discussion on the research background and motivation (section 2). We then formalise the semantic-web software and legacy clinical database integration problem statement (section 3). We discuss certain aspects of the practical implementation in section 4 and related work in section 5. We conclude with general remarks on the proposed solution and future work.

2 Integrating Semantic-Web Technology with Legacy Clinical Databases

2.1 Research Motivation

In [3], we describe a number of serious limitations to the web-application/database architecture of many "traditional" health information management systems. Some of the system limitations we identified included: (i) difficulty in implementing "intelligent" and adaptive behaviour in the applications (ii) difficulty in modifying the applications' workflows without substantial software reengineering work (iii) complexity and lack of flexibility in the management of clinical rules (iv) difficulty in selecting and displaying relevant context-sensitive clinical information (v) lack of support for reusing third party clinical knowledge. Re-engineering HIMS into "knowledge-aware" systems can efficiently address some of the previous shortcomings. In addition, our interpretation of "knowledge-aware" clinical information systems is consistent with a current trend in health services towards increased standardisation and interoperability of information transfer formats, structures and nomenclature and terminologies (e.g. HL7, CDA, SNOMED-CT). To efficiently manage clinical information systems, we have recommended architectures using ontology-driven work-flows and clinical ontologies. Some of the benefits of integrating semantic-web technology in HIMS include:

Information Representation: High-level, persistent semantic information representation, ideally (i.e. if well designed) not dependant on specific implementation choices and details of applications.

Information Interoperability and Reuse: Increased interoperability of information through the use of standard information models or terminologies. In particular, mapping to a single information model itself mapped to other information models can insure widespread interoperability through cross-referencing of information items. This is particularly true of clinical terminologies and a strong argument in favour of using a reference model in clinical information systems.

Information Processing: More powerful information processing functionalities (e.g. *logical reasoning, classification*) than commonly available through other rule engines, which are typically more suited to algorithms and scores calculations.

Although our own experience with designing decision support systems strong-ly highlighted the benefits of using knowledge-bases at the core of the information system architecture, this does not resolve the issue of how to deal with *legacy clinical systems*. In the worst case scenarios, these systems may include clinical databases with no underlying information models whatsoever: these systems were developed as *ad-hoc* solutions in response to short-term requirements and have grown *"organically"* over the years. These legacy systems are not uncommon in the health services and confront designers responsible for upgrading HIMS with non trivial challenges, as upgrade solutions must be performed without a breakdown of service delivery. We next describe our implementation of a pre-operative assessment software incorporating semantic-web technology and then formalise the reverse knowledge engineering problem of integrating the system with a legacy database.

2.2 Research Background

This work is part of an ongoing project to introduce *semantic* technology into a preoperative risk assessment system software called Synopsis. The use of knowledge representation and reasoning both completes functionalities and overcomes a number of limitations of the existing system. The overall architecture of the preoperative assessment system is illustrated in Figure 1. We refer the interested reader to [1,2,3,4] for a detailed description of the system features and functionalities.

Prior to introducing semantic technology within the system, the preoperative software was only composed of the following elements: user input (step 1), clinical data storage (3.a) and rule engine (4.c). Therefore, the preoperative risk assessment (5) was almost entirely based on the calculation of numeric scores. Thus, the introduction of semantic based technology in the system enables adaptive information collection (2a and 2b), high level semantic patient modelling (3.b) and decision support based on classification (4.a and 4.b) rather than numeric rules only. This provides for a significant enhancement to the functionalities and capabilities of the system.

In the system, decision support is usually provided in a 2 step process. The first step typically calculates risk scores or derives risk grades (ASA grades, surgical risk grades, etc.) using numerical formulas such as the Goldman and Detsky cardiac risk index previously mentioned in this article, the Physiological and Operative Severity Score for the enUmeration of Mortality and Morbidity (POSSUM) [5], etc. At this stage, the system does not use the decision support ontology but merely computes values using an open source Java-based rule engine (JBoss Rules[2]). Once the risk grades and categories have been derived from the

[2] http://www.jboss.com/products/rules

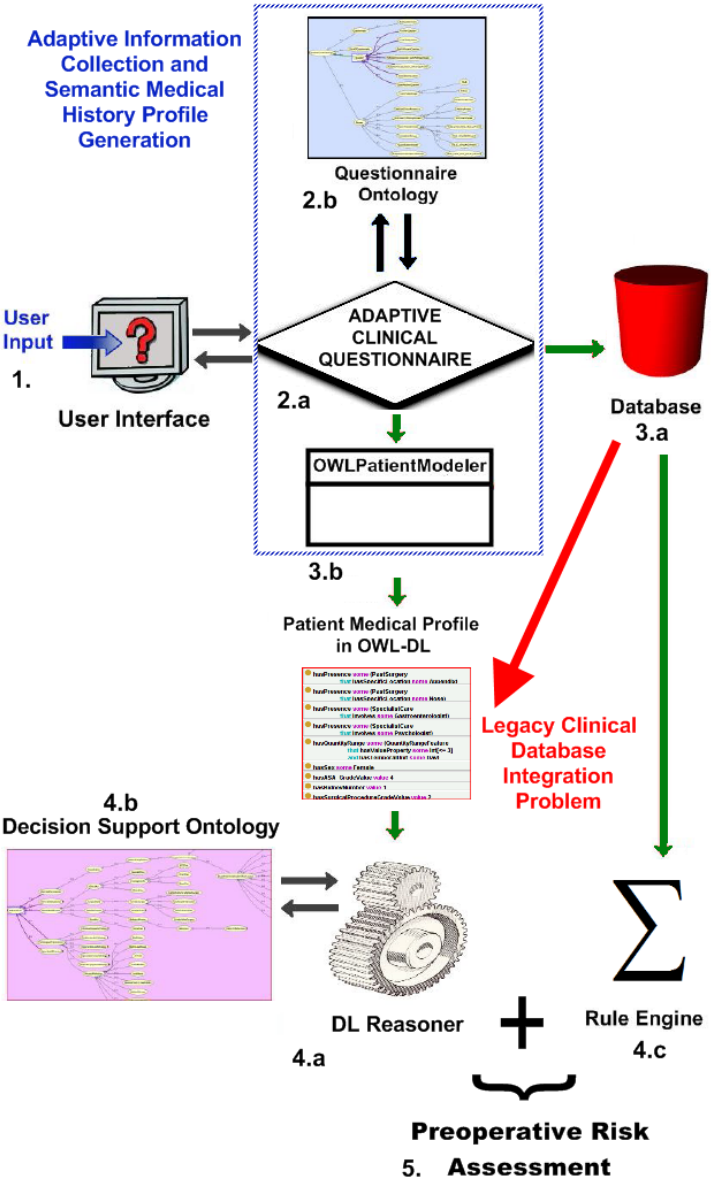

Fig. 1. Architecture of the clinical knowledge-base Preoperative risk assessment system

first risk calculation step, the system then performs decision support using the open-source java-based PELLET reasoner [6] to reason on the decision support ontology given a patient OWL medical history profile. Concrete examples of reasoning with the decision support ontology can be found in [4].

3 Dealing with Legacy Clinical Databases: A Formal Reverse-Knowledge Engineering Problem

3.1 Problem Statement

Figure 1 highlights very clearly the problem which arises when dealing with a legacy clinical database. This is a practical problem for CIS software engineers as the Synopsis software has been in use in a number of hospitals in the UK and Netherlands for several years and the hospitals' HIMS hold several thousands patients preoperative records. The previous version of the information system did not have any of the semantic web technology components described in the previous section.Therefore, the adaptive information collection components of the system (2.a and 2.b) were not previously available. As a consequence, the patient OWL medical history profile generation (3.b) and ontology-driven decision support functionalities (4.a and 4.b) are not currently possible for older patient records (although available to new patients entered in the system). Effectively, the system was only composed of the screening questionnaire input interface (1.), clinical database (3.a) and rule engine (4.c). Thus, in order to use the new semantic-based decision support functionalities, the problem becomes: *"how to generate a high level semantic representation of the patient medical history directly from the low level data representation contained in the database?"*. In other words, the issue - which is represented in the Fig. 1 by the red arrow - sums up as: how to generate the patient OWL medical history (3.b) from the database (3.a) without going through the intermediate semantic generation component (2.a)? This is a reverse engineering problem, which consists of making *explicit* the *implicit* information contained in the database. The scope of the problem is best illustrated with the examples contained in Table 1.

One can clearly see the considerable amount of implicit information contained in the database. In example 1 of Table 1, a database entry with an associated value of 0 means the *absence* (false) of a specific concept (comorbidity) while a value of 1 indicates the presence of this comorbidity (true) (e.g 2). In the examples 3 and 4, these same values of 0 and 1 now take on a different meaning as the presence or absence of a specific concept is additionally *implicitly constrained* by a notion of *truth within a specific time range*. Example 5 shows yet more implicit information in the form of several implicit concepts (SurgicalProcedure, Bleeding, Risk, etc.) and properties (wichConsistsOf, withinVolumeRange) as well as an implicit threshold value, which also needs to be combined with the appropriate unit information for proper interpretation of the medical information. It now takes a complex axiom in OWL to explicitly express all this information implicit in the database (key/value) pair. Finally, e.g. 6 shows how different (key/value) pairs this time affect a "qualifying", or "modular" concept within the OWL axiom. In this case, the degree of severity which may range depending on database value from a LOW to a HIGH severity status.

Effectively, in the absence of an explicit information model or database documentation, it is impossible to understand the meaning of a database entry without looking at a combination of: (i) the information on display on the user

Table 1. Low level (implicit) database information representation vs. High level semantic (explicit) representation

e.g	Database (Key/Value)	Lay meaning	OWL Axiom
1.	(diabetes, 0)	"the patient does NOT have diabetes"	NOT (hasComorbidity some Diabetes)
2.	(diabetes, 1)	"the patient has diabetes"	hasComorbidity some Diabetes
3.	(ecgTest, 0)	"the patient has NOT had an ECG test within the last 6 months"	NOT (hasPresence some (ECGTest and (withinTimeRange some int[≤ "6"] and (hasTemporalUnit some Month))))
4.	(ecgTest, 1)	"the patient has had an ECG test within the last 6 months"	hasPresence some (ECGTest and (withinTimeRange some int[≤ "6"] and (hasTemporalUnit some Month)))
5.	(bloodLoss -Risk, 1)	"the patient is to undergo a surgical procedure with an estimated blood loss risk of less than 500 ml"	hasPlannedSurgicalProcedure some (SurgicalProcedure and (whichHasAssociated some (Risk and (wichConsistsOf some (Bleeding and (withinVolumeRange some int[≤ "500"] and (hasVolumeUnit some Millilitre)))))))
6.	(angina -Pectoris, 3)	"the patient has Angina Pectoris and is affected by a marked limitation of ordinary activity as a result"	hasAssociatedComorbidity some (AnginaPectoris and (whicHasAsConsequence some (PhysicalFitnessStatus and (whichHasAssociated some (Limitation and whichHasSeverity some Severe)))))

interface (e.g. screening questionnaire or text related to data input), (ii) the underlying programming code describing which (key/value) is attached to which data input and, in some cases, (iii) how the (key/value) pairs relate to each other through the rules used in the rule engine. Even then, clearly identifying the meaning of a specific (key/value) pair can remain ambiguous to all but the database administrators. These examples highlight how holding dozen of thousands of patient records in repositories in ad-hoc information format is an unadvisable, although not uncommon, situation for health services institutions. This has clearly negative implications in terms of information sharing and systems interoperability.

3.2 Definitions

As a necessary step to deal with the issues highlighted by the previous examples, we introduce a set of definitions to formalise the reverse knowledge-engineering process of migrating a legacy clinical database to a semantic knowledge-based

information system. We here deliberately attempt to use generic definitions as we consider the problem statement addressed in this paper as a generic reverse-knowledge engineering issue which goes beyond the specific implementation of the information system.

Information System: We abstract an information system as a tuple composed of 5 elements: Information Input (I_{in}), Information Representation (I_{re}), Information Storage (I_{sto}), Information Processing (I_{pro}) and Information Output (I_{out}).

$$\mathcal{IS} = \quad < \mathcal{I}_{in}, \mathcal{I}_{rep}, \mathcal{I}_{sto}, \mathcal{I}_{pro}, \mathcal{I}_{out} > \tag{1}$$

Note that these elements do not necessarily refer to a sequential process, as several of the elements may be involved at various points during the actual operation of an information system (e.g. there may be several information collection phases, various types of computation and output, etc.) The Information Representation element of the information system refers to the use of some persistent form of information encoding in the system (e.g. a table, spreadsheet, XML file, programming code, etc.) The assumption here is that this information, used as a resource at runtime, is embedded within the system at the design and development stage, with perhaps (but not necessarily) the option to later manage and update this information. The Information Input element refers to the process of dynamically including new information in the system with the expectation that this will be used by the system to produce results of interest to the user. An example of information input could be a keyword or a query search, filling in a form or collecting medical history information from a patient. The Information Storage element consists of the mechanisms and formats used to store information collected during the input phase in a persistent form. This can range from a very low level representation such as a coma value separated file, key/values in a database, a structured representation such as XML, or higher level semantic representation perhaps using RDF or OWL. The Information Processing element refers to the computation phase of the system which will likely involve both the information storage and information representation elements of the system. Finally, Information Output refers to the process by which the results of the information processing phase is then consumed by the user (e.g. display on screen, a form printed, an order made, a ticket booked, etc.)

Database: As previously highlighted in section 3.1, we are particularly interested in dealing with the situation where the Information Storage element I_{sto} of the information system $\mathcal{IS}$ consists of a rather low level database representation: a set of (key/value) pairs. Thus, we consider an information system $\mathcal{IS}$ where $I_{sto} = \mathcal{D}$, a database as a finite set of key/value pairs, where the key k_i is a unique identifier (typically a string) associated with a corresponding unique value v_i. The value can be binary (0,1), boolean (true, false), a string (e.g."knee prosthesis"), any kind of (integer or real) number or even an object.

$\mathcal{D} = \{(k_1, v_1), ..., (k_n, v_n)\}$
where $n = |D|$ is the total number of fields in the database.

(Patient) Records: A (patient) record r_i is a subset of key/value pairs in the database $\mathcal{D}$ which refer to a unique and distinct entity of specific interest to the information system (e.g. a patient medical record).

$r_i = \{(k_{i1}, v_{i1}), ..., (k_{ij}, v_{ij})\}$

where $j = argmax|r_i|$ is the maximum size of all potential key/value pairs relating to a single record.

As an example, k_{i1} could refer to the first name of "John Haym", k_{i2} to his surname and k_{i3} to his age of 42 in which case we would have v_{i1}="John", v_{i2}="Haym", v_{i3}=42, while (k_{ij}, v_{ij}) refer to all other information contained in the database relating to the patient "John Haym" (e.g. hospital number, comorbidities, medical history, etc.). Note that some of the values in the records may be missing or unknown.

We call $\mathcal{R}$ the set of all patient records in the database $\mathcal{D}$.

$\mathcal{R} = \{r_1, ..., r_t\}$

where $t = |R|$ is the total number of records in the database. If we call $\mathcal{A}$ (as in *A*dministration) the set of all key/value pairs in $\mathcal{D}$ not directly related to (patient) records, then we have: $\mathcal{D} = \mathcal{R} \cup \mathcal{A}$

Note that if the database $\mathcal{D}$ is uniquely composed of the set of (patient) records $\mathcal{R}$, then $\mathcal{A}$ is the emptyset, $\mathcal{A} = \emptyset$ and $\mathcal{D} = \mathcal{R}$.

(Clinical) Knowledge Base: We define a Knowledge Base $\mathcal{K}B$ as a collection of statements, *sentences* or *axioms* about a specific domain knowledge [7]. To restrict the scope of our knowledge base , we assume that we are interested in constructing $\mathcal{K}B$ as a collection of useful statements of interest to our information system $\mathcal{I}S$. Including $\mathcal{K}B$ in $\mathcal{I}S$ introduces some of the benefits previously highlighted in section 2 of this article: an improved persistent information representation, increased interoperability, new computation functionalities, etc.

Revisting the previous definition of our information system, the information system can thus be transformed into the following tupple:

$$\mathcal{I}S = \quad < \mathcal{I}_{in}, \mathcal{I}'_{rep}, \mathcal{I}'_{sto}, \mathcal{K}B, \mathcal{I}'_{pro}, \mathcal{I}'_{out} > \tag{2}$$

The assumption here is that the introduction of a knowledge base in the information system has no immediate effect on the information input $\mathcal{I}_{in}$, as this is beyond the control of $\mathcal{K}B$. However, we assume that it *may* impact on all the other elements of the information system. As an example, new information processing functionalities $\mathcal{I}'_{pro}$ - and perhaps more expensive computation (e.g. *logical reasoning* for example) - may now be possible within the information system. As a result, additional output $\mathcal{I}'_{out}$ may be possible, more efficient, perhaps more demanding storage $\mathcal{I}'_{sto}$, etc.

3.3 The Information System Reverse-Engineering Integration Problem

Using the previous definitions, the *goal* of the information system reverse engineering integration problem is to migrate the representation of the information system from state (1) to state (2), or to be more specific to generate the knowledge base $\mathcal{K}B$ within state (2), with as little effort as possible, i.e. mostly by automatic means. Thus here lies the first problem:

- as clearly illustrated by the examples in section 3.1, the implicit information of the information system $\mathcal{I}S$ in state (1) is at best spread accross all its constituting elements (i.e. $\mathcal{I}_{in}, \mathcal{I}_{rep}, \mathcal{D}, \mathcal{I}_{pro}, \mathcal{I}_{out}$) (with $\mathcal{D}$ standing for $\mathcal{I}_{sto}$) and at worst, beyond (i.e. external documentation, system designers, database administrators). But...
- automating the knowledge base generation requires some specific input, in our case, the (key/value) pairs contained in the database $\mathcal{D}$.
- thus, to make the knowledge implicit in the database $\mathcal{D}$ explicit, one needs an external source of information, which we call the *axiom Mapping Function* $\mathcal{M}f()$. $\mathcal{M}$f() takes as input a single (value/pair) (k_i, v_i) from the database $\mathcal{D}$ and maps it to a single *axiom* of the knowledge base $\mathcal{K}B$, (see Fig. 2):

$$\forall (k_i, v_i) \in \mathcal{D}, \quad \mathcal{M}f\ (\ (k_i, v_i)\ ) \longmapsto Ax_i,\ \ Ax_i \in \mathcal{K}B$$

- as the mapping function $\mathcal{M}$f() requires access to information external to the database $\mathcal{D}$ (i.e, the implicit information), the process of generating $\mathcal{M}$f() needs to be performed manually by a knowledge engineer. However, the mapping function needs not be defined for all values of $\mathcal{D}$. If $\mathcal{M}$f() is defined for $argmax|r_i|$: all the potential (key/value) pairs of a single patient record, then the process of generating the knowledge base $\mathcal{K}B$ can be automated for *all* patient records $\mathcal{R}$ (see Fig. 3)
- an important practical factor to take into consideration is that, the mapping function $\mathcal{M}$f() might not in practice *need* to map to *all* the $argmax|r_i|$ potential (key/value) pairs of a single patient record, but only to a "useful" *subset*. In fact, it is very likely that the mapping requirement will be considerably less. Effectively, the $\mathcal{M}$f() function only needs to map to what is required for generating a knowledge base sufficient for the purpose of the operation of the semantic component of the information system (see discussion on practical implementation in the next section).

4 Discussion on Legacy Database Integration Implementation

While the integration problem of the legacy database can be described generically, this is not true of the implementation solution which will necessarily depend on a number of factors specific to the information system it is applied to, including any information model behind the database implementation, the extend

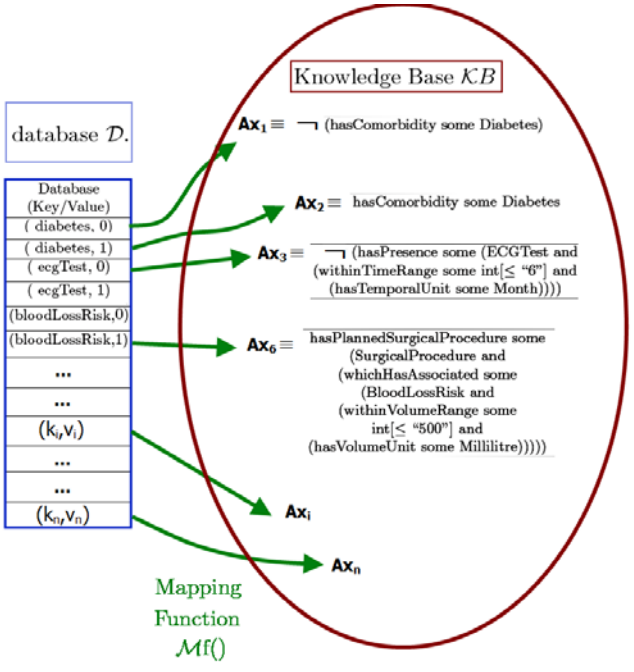

Fig. 2. The axiom Mapping Function $\mathcal{M}$f() from database $\mathcal{D}$ to knowledge base $\mathcal{KB}$

of implicit information in the database, the level of semantic granularity (i.e. complexity of axioms in the knowledge base) required for the effective operation of the semantic application, etc. For illustration purposes, we will provide some hindsight into the migration process of the Synopsis database. It took approximately 8 weeks to an experienced knowledge engineer to (i) design the mapping function $\mathcal{M}$f() for a selected subset of the $argmax|r|$ potential (key/value) pairs of a single patient record r and (ii) write the necessary java OWL API code necessary to automate the mapping from the whole patient database $\mathcal{D}$ to the knowledge base $\mathcal{KB}$. Although this is certainly a non-negligible amount of skilled man-work to migrate the information system database, this was more then compensated in term of cost-benefits by the fact that this provided the upgraded system with backward compatibility to dozens of thousands of patient records held in hospitals databases.

In practice, the $argmax|r_i|$ potential (key/value) pairs of a single patient record proved to be quite high, even given the limited scope of the information in the database (i.e. patient preoperative information as opposed to *full* patient medical record). The reason for the high value of the potential (key/value) pairs is in part explained by: (i) book-keeping purposes which means that even when an information is missing, the information is recorded as (key/ value = null) and (ii) the presence of a number of checklists, each with several dozens items (i.e. all potential allergies, procedures, comorbidities, etc.) The size a real single patient record contained in the database is likely to be a lot smaller, e.g.

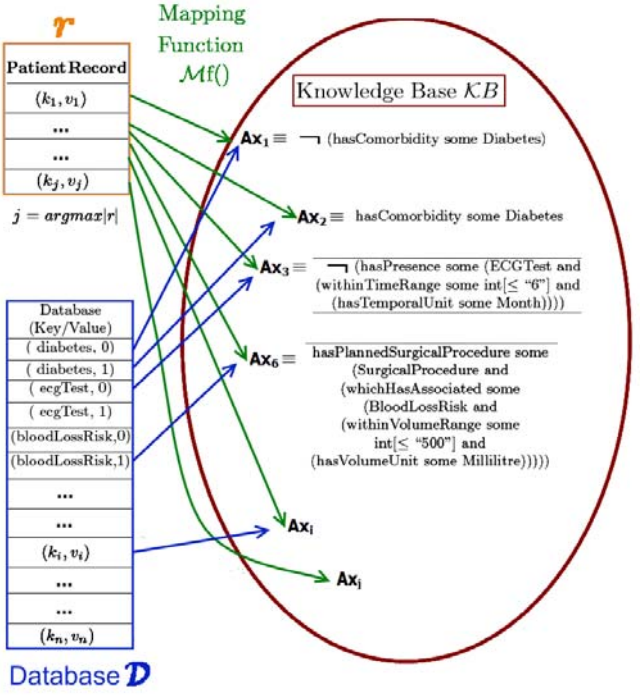

Fig. 3. The Mapping Function $\mathcal{M}$f() only needs to be defined for the $argmax|r|$ potential (key/value) pairs of a single patient record r in order to map the whole patient database $\mathcal{D}$ to the knowledge base $\mathcal{K}B$

around a hundred (key/value) pairs. At the time of writing, $\mathcal{M}$f() translated into the mapping of around 300 axioms in the knowledge base. The reason why the number of axioms mapped in the knowledge base is a lot smaller then the number of all potential (key/value) pairs of a single patient record is that we have prioritised generation of axioms which are directly exploitable by the reasoner-based decision support components of the information system (4.a and 4.b in Figure 1). Remaining unmapped (key/value) pairs are either of the bookkeeping nature previously mentioned which we do not yet exploit in the decision support system or correspond to values which are more effectively handled by the rule engine component of the system (4.c). However, completing the mapping function for all the potential (key/value) pairs of a single patient record will eventually provide the additional benefit of cross-referencing the record to other information models.

5 Related Work

[8] discuss the migration of relational schemas to a knowledge base through the following steps: reverse engineering information capture from database schemas,

applying a set of mapping rules and finally evaluating, refining and validating the mapping. [9] present a non-exhaustive review of various approaches of ontology learning from databases. [10] discuss the issue of describing the semantic relationships between elements of a relational database and a specific ontology. A practical use of developing this correspondence is enabling the formulation of conceptual queries into logical (data level) queries. [11] describe the issues surrounding handling aging legacy information systems (LIS) in business organisations. They highlight that technology oriented solutions present a number of challenges, including: total breakdown of service while attempting to redevelop systems from scratch, short-term fix (e.g. wrapper approach), step by step approach with the risk of ultimately ending with an outdated system by the time the migration is complete, etc. More importantly, they argue that the most important aspect from the business organisation perspective is to preserve the *implicit* business knowledge accumulated in the data within the LIS repository over the years. They stress the importance of capturing this knowledge in a technology-independent way and recommend an *ontological* approach. They suggest a process called *Content Sophistication*, which is conducted in two steps: *Interpretation* (identifying business objects semantics) and *Sophistication* (improving the model by removing discrepancies). [12] propose to handle the issue of data integration across multiple databases by referencing individual databases to a corresponding ontology. The proposed methodology consists of first automatically generating semantic concepts from database schemas. The ontology is then manually annotated by reverse engineering the database relational model. Finally, changes to the database schemas can be automatically propagated to the corresponding ontology using a mapping from database schema change primitives to the ontology. [13] suggest transforming HTML forms of web applications into XML schemas, use these to generate a UML model and finally extract OWL entities. [14] propose a formal framework for extracting ontologies from relational databases. [15] describe DBOM, a java-based framework using a manually engineered XML declarative mapping as a set of explicit correspondences between a database and the knowledge base models.

6 Conclusion

We have discussed the issues surrounding migrating legacy health information management systems into semantic enabled applications. We looked specifically at the issue surrounding the generation of a knowledge-base of high-level semantic information representation in OWL from a database containing a low data level information representation. We have formalised the reverse knowledge-engineering problem of making *implicit* information *explicit* through the semi-automatic generation of a knowledge-base of all patient records. We have discussed a semi-automatic methodology consisting of a 2 step process, including: (i) manually generating an axiom mapping function $\mathcal{M}$f() for a selected useful *subset* of all potential (key/value) pairs of a *single* patient record and (ii) using this mapping to automatically generate the clinical knowledge-base of high

level medical information for *all* patient records. This thus provides an upgraded semantic system with backward compatibility to all patient records held in the legacy information system database. Future work will include evaluation of its operation in field studies at selected pilot sites.

References

1. Bouamrane, M.M., Rector, A., Hurrell, M.: Gathering Precise Patient Medical History with an Ontology-driven Adaptive Questionnaire. In: Proceedings of the 21st IEEE International Symposium on Computer-Based Medical Systems, CBMS 2008, Jyväskylä, Finland, pp. 539–541. IEEE Computer Society, Los Alamitos (2008)
2. Bouamrane, M.-M., Rector, A.L., Hurrell, M.: Ontology-Driven Adaptive Medical Information Collection System. In: An, A., Matwin, S., Raś, Z.W., Ślęzak, D. (eds.) Foundations of Intelligent Systems. LNCS (LNAI), vol. 4994, pp. 574–584. Springer, Heidelberg (2008)
3. Bouamrane, M.M., Rector, A.L., Hurrell, M.: Using Ontologies for an Intelligent Patient Modelling, Adaptation and Management System. In: Meersman, R., Tari, Z. (eds.) OTM 2008, Part II. LNCS, vol. 5332, pp. 1458–1470. Springer, Heidelberg (2008)
4. Bouamrane, M.M., Rector, A., Hurrell, M.: Development of an Ontology of Preoperative Risk Assessment for a Clinical Decision Support System. In: Proceedings of the 22nd IEEE International Symposium on Computer-Based Medical Systems, CBMS 2009, Albuquerque, US. IEEE Computer Society, Los Alamitos (to appear, 2009)
5. Copeland, G., Jones, D., Walters, M.: Possum: a scoring system for surgical audit. British Journal of Surgery 78(3), 355–360 (1991)
6. Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A., Katz, Y.: Pellet: A practical OWL-DL reasoner. Journal of Web Semantic 5(2), 51–53 (2007)
7. Brachman, R., Levesque, H.: Knowledge Representation and Reasoning. Morgan Kaufmann, Elsevier (2004)
8. Stojanovic, N., Stojanovic, L., Volz, R.: A reverse engineering approach for migrating data-intensive web sites to the semantic web. In: Proceedings of Intelligent Information Processing, IFIP 17th World Computer Congress, Montral, Qubec, Canada, August 2002, pp. 141–154 (2002)
9. Gottgtroy, P., Kasabov, N., MacDonell, S.: An ontology engineering approach for knowledge discovery from data in evolving domains. In: Proceedings of Data Mining IV, Rio de Janeiro, Brasil, pp. 43–52 (2003)
10. Verheyden, P., De Bo, J., Meersman, R.: Semantically unlocking database content through ontology-based mediation. In: Bussler, C.J., Tannen, V., Fundulaki, I. (eds.) SWDB 2004. LNCS, vol. 3372, pp. 109–126. Springer, Heidelberg (2005)
11. Daga, A., de Cesare, S., Lycett, M., Partridge, C.: An ontological approach for recovering legacy business content. In: Hawaii International Conference on System Sciences, vol. 8, p. 224a (2005)
12. Kupfer, A., Eckstein, S., Neumann, K., Mathiak, B.: Handling changes of database schemas and corresponding ontologies. In: Roddick, J., Benjamins, V.R., Si-said Cherfi, S., Chiang, R., Claramunt, C., Elmasri, R.A., Grandi, F., Han, H., Hepp, M., Lytras, M.D., Mišić, V.B., Poels, G., Song, I.-Y., Trujillo, J., Vangenot, C. (eds.) ER Workshops 2006. LNCS, vol. 4231, pp. 227–236. Springer, Heidelberg (2006)

13. Benslimane, S.M., Malki, M., Rahmouni, M.K., Benslimane, D.: Extracting Personalised Ontology from Data-Intensive Web Application: an HTML Forms-Based Reverse Engineering Approach. Informatica 18(4), 511–534 (2007)
14. Lubyte, L., Tessaris, S.: Extracting ontologies from relational databases. In: Proceedings of the 20th Int. Workshop on Description Logics, DL 2007, Brixen-Bressanone, Italy, pp. 387–395 (2007)
15. Cure, O., Bensaid, J.D.: Integration of relational databases into OWL knowledge bases: demonstration of the DBOM system. In: Proceedings of the 24th International Conference on Data Engineering Workshops, ICDE 2008, Cancn, Mxico, April 2008, pp. 230–233. IEEE Computer Society, Los Alamitos (2008)