Implementation of Ontology for Intelligent Hospital Wards

Pavandeep Kataria, Radmila Juric, Shamimabi Paurobally, Kambiz Madani
School of Informatics, University of Westminster
115 New Cavendish Street, London W1W 6UW, UK
E-mail {P.Kataria1@wmin.ac.uk; R.Juric@wmin.ac.uk; S.Paurobally@wmin.ac.uk; K.Madani@wmin.ac.uk }

Abstract

We have developed and implemented an ontology for an intelligent hospital ward. Our aim is to address the pervasiveness of computing applications in healthcare environments, which require: sharing of data across the hospital, including data generated by sensors and embedded in such environments, and dealing with semantic heterogeneity that exists across the hospital's data repositories. Our conceptual ontological model that supports such an environment has been implemented using semantic web tools and tested through the application developed with the J2EE technology.

1. Introduction

Healthcare software applications have changed significantly in the last decade. The advances of wireless and mobile computing and proliferations of pervasive healthcare technologies with heterogeneous embedded devices have made a huge impact on how we develop computing environments in healthcare. Hospital software systems specifically require merging traditional and pervasive computing applications and are expected to deal with an enormous amount of information and data stored in a variety of forms: from structured patient electronic records to multimedia data streams of medical images. They should also exhibit high quality multimedia and context aware communications and computations. Moreover, hospital operational environments depend on collaboration between different specialists and exchange of their expertise, and thus sharing of knowledge, information and ultimately patient clinical data amongst software applications is essential. Heterogeneities of hardware/software platforms and software applications that are affected by constant technological changes are unavoidable in healthcare. Pervasive computing enables the development of ubiquitous healthcare software environments, which extends the traditional delivery of healthcare in primary care and hospitals, towards personalized self-care, home care or any other healthcare service which can be delivered at any time and any place.

In our works we address some of the issues above by designing a computational environment which fits within a typical hospital ward and which deals with:
(i) sharing of data across wards, hospital departments and administration,
(ii) alleviating the interoperability problem that arises from semantic heterogeneities, which might exist across the applications and data repositories that reside within ward, hospital departments and administration and
(iii) capturing "context awareness", which is based on the use of data generated from embedded devices, which affect the software application's behavior.

We aim to address all three: data sharing, semantic interoperability and context awareness, by building a software application for a hospital, which is based on an ontology that makes provisions for (i)-(iii). The term "intelligent" is used to refer specifically to the role of ontologies in solving the problems from (i)-(iii). This adds to a variety of definitions on what an intelligent hospital environment is: from being supported by context and location aware software applications [1] and enabling automation of adaptive workflows of clinical processes [2] to having various patient medical assistants, which improve patient care and safety in hospital settings, such as in [3].

We design and implement a Hospital Intelligent Ward Ontology (HIWO), as a formal description of an intelligent hospital domain. We give its explicit concepts and the relationships behind them. HIWO provides a common understanding of the hospital environment for its domain users, thus enabling the sharing of data and capturing context awareness, as suggested in (i)-(iii) above.

1530-1605/08 $25.00 © 2008 IEEE

In section 2, we give an overview of related works through examples of hospital software systems that either use ontologies or address the issues of data sharing, interoperability and context awareness. In section 3, we give an example scenario of an intelligent hospital ward and we illustrate the functionality of a software application that supports it. We also explain how such an application would interpret and implement data sharing, interoperability and context awareness. In section 4, we describe the design and implementation of HIWO. We illustrate a few classes and properties of HIWO and outline their implementation in OWL. We show steps of the HIWO implementation through importation, integration and exportation. We also report on the choice of semantic web tools, which enabled us to implement HIWO using OWL. In section 5, we describe an excerpt of the EJB application prototype for testing the validity of HIWO. Section 6 concludes and lists our future works.

2. Related Works

It has been difficult to find any similar work, which contributes towards intelligent hospital environments, by addressing all three issues: data sharing, interoperability and context awareness. The most relevant works on sharing of medical data for faster and more effective delivery of healthcare are: works of T. Ueckert et al. from [4] which look at the adaptive workflows within hospitals that lead towards intelligent automation in healthcare organizations, works of J. Sutherland et al. from [2] and S. Mitchell et al. from [5], which investigate context aware mobile communications in hospitals to improve the management and sharing of information, or works which support personalization of healthcare services in context aware healthcare systems [6], to mention just a few. T.R. Hansen et al. in [7] describe an interactive hospital scheduling and awareness system, which supports intense coordination of operations in a large hospital and incorporates location tracking, context awareness system and interactive user interfaces to ensure quality of communications within the hospital. They focus on the technological aspects of designing and implementing such applications, with no information on how data and context repositories are managed and shared, and whether they use an ontology for modelling contexts and addressing data sharing and their heterogeneities or not. J.E. Bardram, and H.B. Christensen also assess and use technologies in pervasive hospital environments in [8], but they design an "activity based computing architecture" which supports the collaborative, nomadic and specialised nature of medical treatments. S. Mitchell et al. in [1] are concerned with multimedia context applications in hospitals, which use their QoS DREAM platform to provide a seamless, context sensitive communication within the hospital, which can adapt to a patient's changing location.

In recent years, ontologies have become central to many applications such as scientific knowledge portals, information management, electronic commerce, and semantic web services. The key purpose and examples of using onotlogies have been elaborated in various works, such as [9], [10], [11], [12], [13], [14], [15], [16] etc., thus providing:

- A common understanding of the structure of the information among different people and end users;
- A formative well defined declarative semantic as a means for supporting the identification of requirements and specification of sub systems;
- Interoperability between data repositories and devices to provide a form of "serendipitous interoperability";
- Reuse of knowledge within an ontology to support the sharing of application specific knowledge.

From this perspective, an ontological approach appears to be ideal for dealing with our aims from (i)-(iii) in section 1.

Thus the use of ontologies in modeling context aware systems is predominant today. There are many examples where ontological models are essential for "context awareness" and knowledge sharing. There are a few examples of "smart environments" in people's homes, which are created with the context technology supported by ontologies, where patients can be monitored remotely, thus enabling mobile and remote delivery of healthcare services [17], [18], [9]. Ontologies are also used to support medical terminology and coding systems, such as [19]. One of the rare examples of an ontology in a hospital scenario is OntHoS [20]. It models scenarios of hospital logistics for the development of clinical information systems and hospital simulation models.

3. The Intelligent Hospital Ward

3.1. The Scenario

We consider a hospital, which consists of various departments, wards, and central administration. Each ward (WARD) stores information about the patients and treatments given to them. They also store patient medical history and data supplied by sensors that (a) monitor patient vital signs of life, such as blood pressure, heartbeat and temperature, (b) send patient location and (c) send patient identification through wearable tags located on the patient's clothes. Each department (DEPT) stores laboratory test and investigations that have been carried out on each

patient, such as blood test, X-RAYs etc. The central administration (ADMIN) stores data on patients and the costs incurred while they are hospitalized. Thus the associated databases contain the following: the WARD database contains the data on patients P_1, … P_n; treatments $T_1 \ldots T_n$, monitored vitals signs VS_1, VS_2, VS_3 and data sent by sensors with patient location S_{LOC} and patient identification S_{PID}. DEPT database contains data on patients P_1 … P_n and data on laboratory tests or any other investigation carried out on patients $TL_1 \ldots TL_n$. The ADMIN database contains data on patients P_1 … P_n; costs associated with treatments and patients in the hospital C_1 … C_n; and data on departments D_1 … D_n and wards W_1 … W_n which is relevant for the central administration.

A nurse N_i and a doctor Dr_i make decisions based on information stored in databases WARD and DEPT. When a specific situation occurs, e.g. when a patient's heartbeat changes or temperature rises, the nurse N_i will be alerted and action should be taken immediately. This might result in taking patient P_i to department D_j where a set of lab tests $TL_1 \ldots TL_n$ will be carried out. As soon as the patient P_i has been moved from ward W_i into Department D_j for the tests, the WARD and DEPT databases are updated, because the patient location S_{LOC} changes. The results of tests are input into DEPT database and are accessible by nurse N_i and doctor Dr_i. They can immediately decide on further treatment $TL_1 \ldots TL_n$ or a change of patient P_i's current treatment(s) T_i. When the Central Administrator CA_i calculates the cost, he/she will consult information from ward W_i on the patient's treatments T_i stored in the WARD database, information from lab test TL_i from department D_j stored in the DEPT database and their own data from the ADMIN database, which should contain the date when the Patient P_i entered the hospital and when he/she could be discharged.

3.2 Examples of Software Functionality

The software applications within the hospital run on network nodes and access any of the existing databases (WARD, DEPT, ADMIN) regardless of the location where each application resides. This means that each operational area: Ward, Department and Administration is responsible for its local database: WARD, DEPT or ADMIN respectively and can access any other database, which is not local to them. The users of these software applications are nurses, doctors, medical staff in various departments and administration. Apart from having continuous information on a patient's vital signs, medication, treatments, laboratory tests and similar, the authorized users of software applications which run within each WARD know at any time the answers to the key competence questions such as: Which treatment is allocated to which patient and at which exact time slot? Which treatment has the patient had or will be having, as a result of information gathered from vital signs, medical history or laboratory tests? Which laboratory test(s) is the patient is having/will have, as a result of changes in diagnosis, changes in vital signs values or treatment changes? At which department are/will be the tests carried out and who is/will be the medical staff responsible for the test? Which cost/s is assigned to each patient at any time? Figure 1 illustrates the scenario of the intelligent hospital ward.

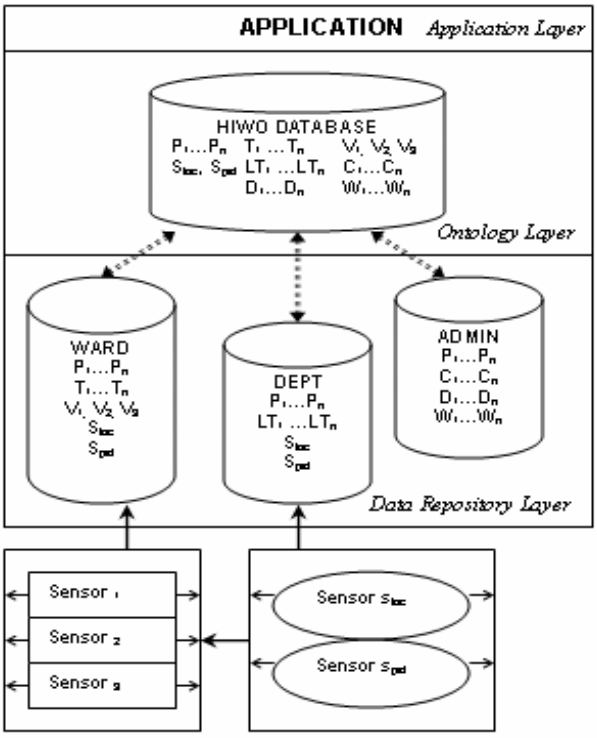

Figure 1: Scenario of the intelligent hospital ward

3.3. Data sharing, interoperability and context awareness

Data sharing (issue (i) in section 1) would mean that if we, for example, stored a patient diagnosis and medical history within the WARD database, these data items should be visible and accessible by the authorized users/ applications local to the Dept operational area, when certain tests are being carried out. Furthermore, when the results of tests are entered within the DEPT database, the application, which runs within the Ward operational environment, KNOWS where the test results are, i.e. it knows where to retrieve data from. It should be able instantly to access test results, thus nurses and doctors who work in that

ward could make decisions on the patient's diagnosis, treatments etc. Data sharing would eliminate "sending" the data items to applications that need them, but would allow the same applications to access and use them regardless of their location.

Context awareness (issue (iii) in section 1) would mean that the applications, which reside within the Ward operating environment, would be aware at any time of the values of a patient's vital signs and the location. Thus context data are collected, interpreted and possibly stored within their own CONTEXT database. When a patient moves, then the values of some context data changes and the action, which results from such changes, may be triggered from either Ward or Dept operational environments. This means that *context awareness* in our applications may require database application triggers to be invoked as a result of context changes. For example: if a patient moves from his/her Ward to a blood test Dept, his/her location changes, thus

- the application which runs within the Ward environment KNOWS that the patient is having a blood test done, and
- the application which runs within the Dept environment KNOWS that blood test results are expected to be entered.

Interoperability (issue (ii) in section 1) would mean that a variety of heterogeneities that exist within the hospital applications are not an obstacle for data sharing and implementing context awareness. Data stored in ADMIN, DEPT and WARD databases may exhibit semantic heterogeneities (platform and systems heterogeneities are outside our scope). Thus they might have heterogeneous DB models and data modalities. For example Patient records stored in the WARD database could have different record structure than Patient records stored in the DEPT database, because: (a) The WARD database may never store multimedia data, but the DEPT database may store X-Rays or any similar data types and (b) Patient records within the WARD database should store a patient's medical history and treatments, which will not be replicated across the ADMIN and DEPT databases, even if they store patient data specific for their applications.

Accessing patient records stored in the DEPT database from a Ward Application would mean that the SQL issued by the Ward Application for the retrieval of patient records from the DEPT database is "correct". Thus the Ward Application's SQL is adequate for retrieving data, which are stored within the DEPT database.

4. The Ontology

The HIWO conceptualizes the scenario from section 3.1, supports the sharing of knowledge, and interoperability between underlying data repositories, and then uses the context based information, as in (i) - (iii) from section 1.

In this section we give a description of the way we modeled and implemented HIWO. We have followed a comprehensive modeling guide using the Web Ontology Language [21] and practical guide for developing OWL ontologies, given in [22]. We have chosen OWL instead of RDFS and DAML+OIL, because of its expressiveness and its recent merging with DAML+OIL. Thus our terminology is based on a practical guide to building OWL ontologies from [23].

4.1. Classes, properties and relationships

The HIWO model has been created by: (1) determining *the domain* and scope of the ontology, (2) defining *the class* and *class nesting levels* and (3) defining the *class properties* and *internal taxonomy structure.*

(1) *The domain* of the HIWO conceptual model was primarily determined by the requirements from the scenario in section 3.1 and a few competency questions listed at the end of section 3.2.

(2) The HIWO *classes* were discovered intuitively, i.e. using our own personal view of the domain. Figure 2 gives a partial definition of the nested levels defined for the LABTEST class hierarchy of HIWO. Their structures are self explanatory.

(3) The *taxonomical relationship* behind each nested level in Figure 2 is organized using the 'is kind of' concept. For instance, if a class A (LABTEST) is a super class and has a sub class of B (LABTEST LOCATION), then every instance of B is also an instance of A. Thus, the different levels of the LABTEST class taxonomy are:

- the *LABTEST* class as the most general concept,
- the *lab test name, is part of lab test type, results, medicine, is part of medicine type* and *date* as the general top level concepts
- *located in ward* and *treatment name* as the bottom most level concepts.

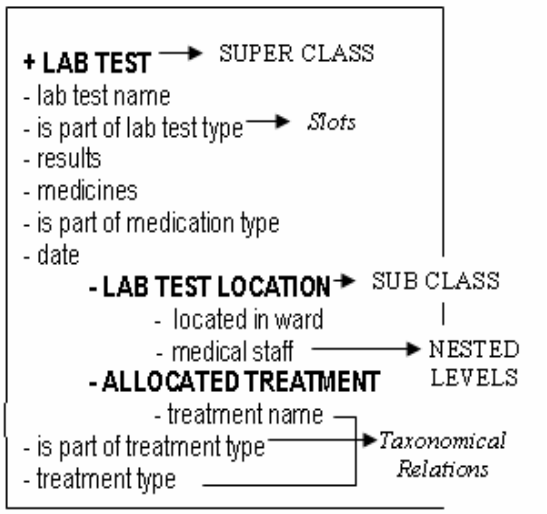

Figure 2 Nested Level in the Taxonomical Relations

Figure 3 also explains how the individual instances of properties from the PATIENT class, can be used as a direct relationship with other classes and their associated properties. In other words, the sub class MEDICAL HISTORY of the super class PATIENT is directly related to the TREATMENT CLASS through using the instances of the property 'previous treatment', i.e. '*Therapy*' to directly create a relationship with the TREATMENT class. More explanations can be found in [24].

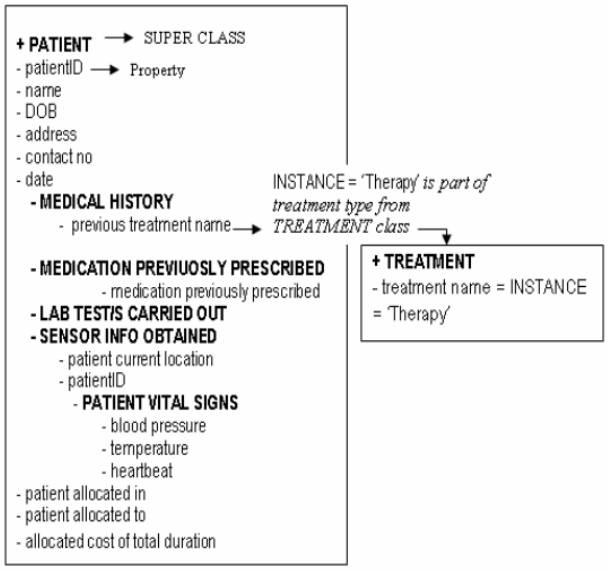

Figure 3: Relationships between Classes and their Properties

4.2. Implementing classes and properties in OWL

When implementing HIWO as an OWL ontology, several modeling constructs within the HIWO ontological model had to be changed. OWL uses properties of a class to create various restrictions based upon the taxonomical relationships defined between classes and their associated properties.

The most common modeling construct which had to change, is the way we defined relationships between classes and their associated properties. We demonstrate this in Figure 4 where the 'PATIENT' class is the *domain* for its associated property 'previous treatment name'. This means that PATIENT class is also a super class in OWL. However, the range of the 'previous treatment name' property is actually the *range* for the property 'treatment name' in the 'TREATMENT' class. Subsequently, the domain for the 'treatment name' property is the 'TREATMENT' class. Thus Figure 4 introduces an example of restrictions dictated by OWL, which must be imposed on our ontological model, which was initially modeled as a relationship in Figure 3.

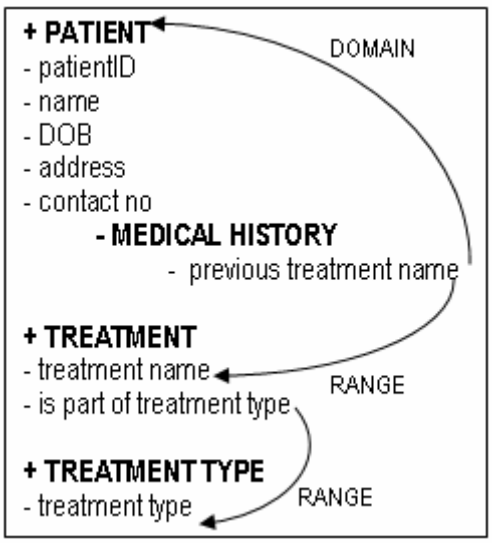

Figure 4: OWL based Relationships Between Classes and their Properties

4.3. Mapping Properties in OWL Classes to the Databases

The prerequisite for the implementation and evaluation of the HIWO ontological model, is to make a proper choice of a technology, which would allow connections to underlying relational databases

(WARD, DEPT, ADMIN). This means that we need a tool, which enables each value of a tuple in an underlying relational database to represent a property instance of an OWL class. In terms of the HIWO ontology, the data stored in WARD database (PatientID), DEPT database (WardID and StaffID) etc. would in turn represent the instance values for the properties defined in our OWL classes for the HIWO ontological model.

The technologies that would assist us in the above are:
i) the Jena API [25] that allows connection to backend relational underlying databases and
ii) the D2RQ API mapping tool [26] that allows automatic mapping of the underlying relational databases; WARD, DEPT and ADMIN to predefined OWL based schemas.

Both technologies are based on Java as they are plug-ins of the .jar archives in the Java library directory. TopBraid Composer [27] which replaces tools in i) and ii) above has become our obvious choice for implementing the HIWO model. It automatically provides a graphical user interface for editing any chosen ontology, provides extensive support for OWL, RDF schema, as well as its integration with Jena and D2RQ plug-ins. However, using the TopBraid Composer has been far from an easy task, due to the lack of tutorials and published examples, which could help us in the HIWO implementation and synchronisation of various versions of plug-ins essential for running the tool.

Figure 5 shows the step-by-step process of HIWO implementation: importation, integration and exportation.

The first step is to map the underlying database schemas (WARD, DEPT and ADMIN) to OWL classes and their associated properties. Our underlying database schemas were created in Oracle 10gi Express Edition [28]. A JDBC connection ensured that each database schema could be imported into TopBraid Composer, thus we perform "importation". TopBraid Composer also allows the use of the D2RQ Mapping Wizard to start the automatic mapping of the relational database schema (WARD, DEPT and ADMIN) into predefined OWL schemas.

The predefined OWL schema created through the mapping represents all converted OWL classes and properties upon each relational table of WARD, DEPT and ADMIN database. This is shown in Figure 6. Each tuple of each table is an OWL property within the OWL class. D2RQ maps primary keys as domain restrictions and foreign keys as range restrictions, thus allowing OWL classes to be created and expressed in accordance to the web ontology language.

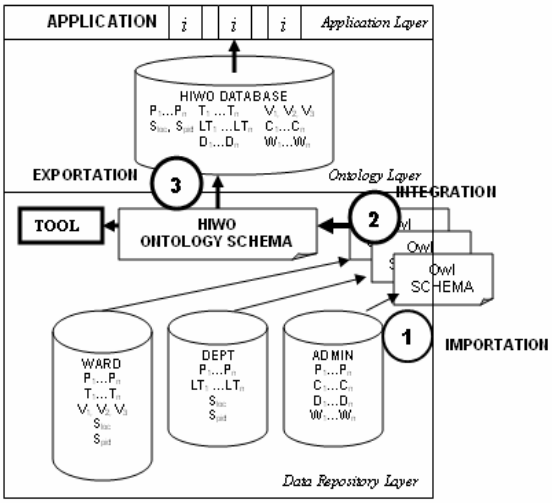

Figure 5: The Implementation Process for HIWO

The second step is the creation of the OWL based HIWO ontological model from section 4.1. Thus we *integrate* the predefined OWL schemas from the automated mapping using D2RQ. We first select the relevant properties needed from our predefined OWL schemas that subsequently represent our OWL properties in the HIWO ontological model. As TopBraid Composer enables users to use different workspaces to import and use a combination of more than one ontology, TopBraid allows us to select properties from predefined OWL schema and store them into a virtual basket workspace (as shown in Figure 7). It converts each retrieved property into a RDF namespace. Using the RDF namespaces as pointers to OWL property instances from the predefined OWL schemas, we are capable of specifying each property in our OWL Classes for HIWO as a specific type called "object property" in OWL.

The final step is the "exportation" of HIWO into persistence storage. As Oracle 10gi Express edition supports RDF spatial namespacing, the exportation is done through a similar process to the "importation". We use the same JDBC connection thus Oracle Express Edition 10gi creates a set of automated tables using "triple stores" to convert OWL classes to RDF tuple storage parameters.

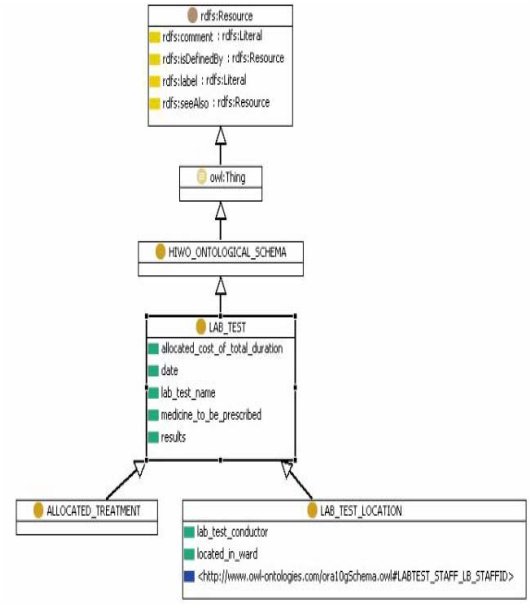

Figure 6: OWL Classes generated from the Underlying Databases

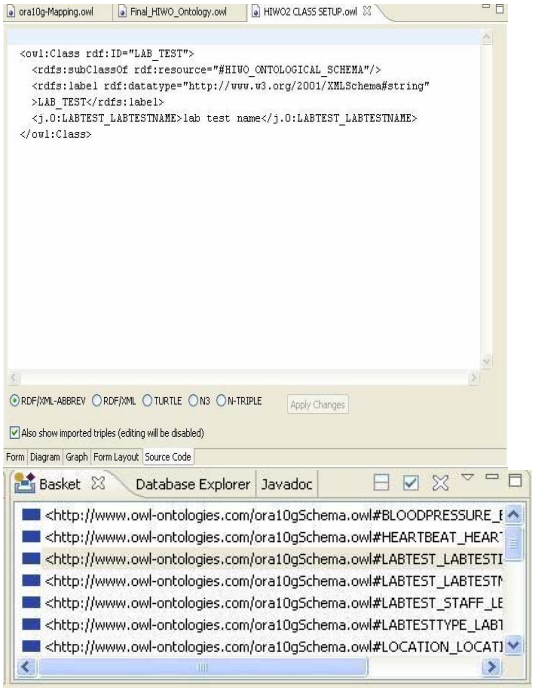

Figure 7: Converting OWL properties into an RDF namespace

5. The Ward Application

We have designed and implemented an EJB application prototype to test and evaluate the HIWO.

We have prototyped the WARD and DEPT databases as relational schemas, which exhibit semantic heterogeneity of patient records and store patient vital signs and locations. Tables and their attributes from the WARD and DEPT database schemas resemble classes and relationships available in Figures 2 and 3. Therefore, ontological classes and relationships mirror the semantics stored in their underlying databases (WARD and DEPT). An EJB application is built upon both: the HIWO database and its underlying databases.

The EJB application components have been modeled and deployed as JSPs, servlets, session and entity beans. We have used the Eclipse tool with the JBOSS application server, because we could plug in the TopBraid Composer into Eclipse. The <<Front.Controller.Servlet>> implements the functionality of accepting the user's input from JSPs and controlling the execution of

(a) Aggregating information relevant for the ward,

(b) Collecting alerts for monitoring patients vital signs issued in a certain period of time and

(c) Retrieving data for a chosen patient across the hospital databases.

Thus the <<Front.Controller.Servlet>> calls <<Aggregate.SessionBean>>, <<Alert.SessionBean>> and <<Retrieve.SessionBean>> components to implement (a)-(c) respectively. Each of these session beans uses a set of entity beans to retrieve relevant data. We have tested retrievals across data stored in hospital databases, through the implementation of the <<Retrieve.SessionBean>> component. The coding was straightforward and fairly automated through Eclipse. The only obstacle was the incompatibility of the tool and their plug-ins when coding and deploying application components, which deserves separate attention. The same has been experienced when implementing the HIWO with the TopBraid Composer.

5. Conclusions

In this paper we describe the HIWO, a hospital intelligent ward ontology, which addresses the problem of data sharing, heterogeneity of data repositories and context awareness in its supporting software applications. We have used the OWL and semantic

web tools for designing and implementing the HIWO and EJB software components for implementing a prototype, which demonstrates the HIWO's feasibility.

Our prototype is relatively simple in order to prove our concept that ontologies are powerful enough to address the issues of data sharing, interoperability and context awareness at the same time, i.e. within the same ontological model. We have also simplified the types and number of contexts in this work, by enumerating them in advance and limiting them to the patient's location and his/her vital signs, which are being monitored. Thus our current work includes the development of (1) the full ontological contextual model, which would not take enumerated contexts and (2) an adequate context management system in terms of enabling context acquisition, collection, interpretation and storage within a context database. This would make our HIWO more powerful and scalable for accommodating changes in technologies and application requirements in hospital environments. However, the software application built upon HIWO is not exclusively a context aware application, because it integrates the traditional hospital software applications that rely solely on relation databases with the pervasiveness of hospital environments. Thus we could currently omit context reasoning and knowledge inferring within our current software application model.

We single out the two sets of problems, which are outside the scope of the paper, but which have affected the implementation of HIWO. The first set is: (a) making the correct choice of semantic web tools for the HIWO's implementation and (b) incorporating the chosen tool's plug-ins which were essential for it's functionality. Both of these are the subject of our immediate attention. Furthermore, the lack of running examples of OWL ontologies in hospital environments that we could relate to, the lack of examples of running applications built upon OWL ontologies and the lack of tutorials, which could assist in learning how to use effectively semantic web tools also affected our work. Thus semantic web tools and technologies still need to mature in order to be used more effectively. We would like to experiment with the development of ontologies without using an ontology editing tool in order to gain more flexibility in ontology modeling and its importations/exportations.

In the near future we expect to have a full-scale implementation of the intelligent ward application and its evaluation within a small private clinic.

6. References

[1] S. Mitchell, M.D. Spiteri, J. Bates, and G. Coulouris, 2000, "Context-Aware Multimedia computing in the Intelligent Hospital", *Proceedings of the Ninth ACM SIGOPS European Workshop. SIGOPS EW2000,*, Kolding, Denmark, September 17, pp. 13-18.

[2] J. Sutherland, and W. Van den Heuvel, 2006, "Towards an Intelligent Hospital Environment: Adaptive Workflow in the OR of the Future," Proceedings of the 39 Annual Hawaii International Conference on System Science, HICSS '06, Kauai, January 4-11.

[3] V.L. Payne and D.P. Metzler, 2005, "Hospital Care Watch (HCW): An Ontology and Rule-Based Intelligent Patient Management Assistant", Proceedings of the 18th IEEE Symposium on Computer-Based Medical Systems, CBMS 2005, Trinity College Dublin, Ireland, June 23-24, pp. 479 – 484.

[4] T. Ueckert, M. Goerz, M. Atain, S. Tessman, and H.U. Prokosch, 2003, "Empowerment of patients and communication with health care professionals through an electronic health record". International Journal of Medical Informatics, 2003 July; 70(2-3), pp. 99-108.

[5] M.A. Munoz, M. Rodriguez, J. Favela, A. Martinez-Garcia, and V. Gozalez, 2003, "Context Aware Mobile Communication in Hospitals", IEEE Pervasive Computing, Vol. 36, Issue 9, pp. 38 – 46.

[6] C.Y. Chin, Z. Yu, and D.Q. Zhang, 2004, "Context-Aware Infrastructure for Personalized Healthcare," in the 2004 International Workshop on Personalized Health, Belfast, UK, December 13-15.

[7] T.R. Hansen, J.E. Bardram, and S. Mads, 2006, "Moving Out of the Lab: Deploying Pervasive Technologies in a Hospital," IEEE Pervasive Computing, Vol. 5, No. 3, pp. 24-31.

[8] J.E. Bardram, and H.B. Christensen, 2007, "Pervasive Computing Support for Hospitals: An overview of the Activity-Based Computing Project," IEEE Pervasive Computing, Vol. 16, No. 1, pp. 44-51.

[9] X. Wang, J.S. Don, C.Y. Chin, S.R. Hettiarachchi, D. Zhang, 2004, "Semantic Space: An Infrastructure for Smart Spaces", IEEE Pervasive Computing, Vol. 3, No. 3, pp. 32 – 39.

[10] H. Chen, T. Finin, and A. Joshi. 2003, "Semantic Web in a Pervasive Context Aware Architecture," Proceedings of the Artificial Intelligence in Mobile System Conference, AIMS 2003, Seattle, USA, October 12-15, pp. 33-40.

[11] H. Chen, T. Finin, and A. Joshi. 2003, "An ontology for context-Aware Pervasive Computing Environments," The Knowledge Engineering Review, Vol. 18, No. 3, pp. 197-207.

[12] H. Alani, S. Kim, D.E. Millard, M.J. Weal, W. Hall, P.H. Lewis, and N.R. Shadbolt, 2003, "Automatic Ontology-Based Knowledge Extraction from Web Documents", IEEE Intelligent Systems, Vol. 10, No 1, pp. 14-21.

[13] D. Oberle, A. Eberhart, S. Staab, R. Volz, 2004, "Developing and Managing Software Components in an Ontology-based Application Server", In Hans-Arno Jacobsen (ed.), Middleware 2004, ACM/IFIP/USENIX 5th International Middleware Conference, Toronto, Ontario, Canada, Vol. 3231 of LNCS, pp. 459-478.

[14] T. Strang, C. Linnhoff-Popien, and K. Frank, 2003, "Applications of a Context Ontology Language" Proceedings of International Conference on Software, Telecommunications and Computer Networks, SoftCom 2003, Split, Croatia, October 10-13, pp 14-18.

[15] T. Gu, D.Q. Xhang, 2004, "Towards an OSGI based Infrastructure for Context aware Applications", in IEEE Pervasive Computing, Vol. 3, Issue 4, pp. 66-74.

[16] L. Razmerita, A. Angehrn, A. Maedche, 2003, "Ontology-based User Modeling for Knowledge Management Systems" Proceedings of 9th International Conference on User Modeling, UM 2003, Johnstown, PA, USA, June 22-26, pp. 172-181.

[17] S. Meyer, and A. Rakotonirainy, 2003, "A Survey of Research on Context-Aware Homes," Proceedings of the 2003 Workshop on Wearable, Invisible, Context-Aware, Ambient, Pervasive and Ubiquitous Computing, Adelaide, Australia, February 7, pp. 159-168.

[18] F.S. Fook, V.S.C., Tay, M. Jayachandran, J. Biswas. D. Zhang, 2006, "An Ontology-based Context Model in Monitoring and Handling Agitation Behavior for Persons with Dementia", Proceedings of the Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops, PERCOMW, Pisa, Italy, March 13-17, pp. 560-565.

[19] A. Rector, B. Smith, W. Ceusters, B. Klagges, J. Köhler, A. Kumar, J. Lomax, C. Mungall, F. Neuhaus, and . Rosse, 2004, "Relations in Biomedical Ontologies," Genome Biology, Vol 6, Issue 5, article R46.

[20] M. Becker, C. Heine, R. Herrler and K-H Krempels, 2002, "OntHoS an Ontology for Hospital Scenarios**"** In: Moreno, A.; Nealon, J. (Eds.): *Applications of Software Agent Technology in the HealthCare Domain*, Whitestein Series in Software Agent Technologies (WSSAT), Birkhäuser Verlag, Basel, pp. 87-103.

[21] D. Brickley, RV.RDF Vocabulary description language 1.0: RDF Schema. World Wide Web Consortium, available at http://www.w3.org/TR/rdf-schema/ (accessed June 2007).

[22] [W3C; OWL] WORLD WIDE WEB CONSORTIUM, 2004, OWL Web Ontology Overview/Guide available at http://www.w3.org/TR/owl-features/ (accessed June 2007).

[23] M. Horridge, H. Knublauch, A. Rector, R. Stevens, and C. Wroe, 2004., 1996, "A Practical Guide To Building OWL Ontologies Using The Protégé-OWL Plugin and CO-ODE Tools" Edition 1.0, available at http://www.co-ode.org/resources/tutorials/ProtegeOWLTutorial.pdf (accessed June 2007)

[24] P. Kataria, R. Juric, K. Madani, J. Croft,. 2007, "Building an Ontology for Intelligent Software Applications in Hospitals", in Proceedings of the 10th world conference on Integrated Design and Processing Technology, IDPT 2007, Antalya, Turkey, June 3-8, pp. 280-286.

[25] [SEMANTIC WEB FRAMEWORK: JENA] JENA – A semantic Web Framework for Java, available at http://jena.sourceforge.net/, (accessed June 2007).

[26] [CHRIS BIZER: D2QR] D2QR – Publishing Relational Databases on the Semantic Web, available at http://sites.wiwiss.fu-berlin.de/suhl/bizer/d2r-server/#about, (accessed June 2007).

[27] [TOPBRAID COMPOSER 2007] TOPBRAID COMPOSER 2007 Features and getting Started Guide Version 1.0, created by TopQuadrant, US, available at http://www.topbraidcomposer.com/, (accessed June 2007).

[28] [ORACLE; ORACLE DB 10G] ORACLE (2007) Oracle 10g Download Page available at http://www.oracle.com/technology/software/products/database/oracle10g/index.html (accessed June 2007).