9th International Conference on DEVELOPMENT AND APPLICATION SYSTEMS, Suceava, Romania, May 22-24, 2008

Dynamically Integrating Knowledge in Applications. An Online Scoring Engine Architecture

Diana GOREA
"Al. I. Cuza" University of Iasi Department of Computer Science
str.G-ral Berthelot nr.16, RO-700418 Iasi
dgorea@infoiasi.ro

Abstract—**The paper presents an method for dynamically integrating knowledge capabilities into applications.The method consists in the applications cooperating with a dedicated system that provides knowledge via Web Services. We propose such a system, called DeVisa, which collects prediction models from one or more producers and provides prediction services to consumers. The prediction services are further used in decision making or business intelligence processes within the consumer applications.**

Index Terms—**data mining, PMML, knowledge, scoring engine, XML native databases, web services**

I. INTRODUCTION

The Knowledge Discovery in Databases (KDD) and in particular data mining component is an important technology for business intelligence because it automates the organizational knowledge acquisition from data resources. The high-level knowledge extracted from low-level data is increasingly seen as a key to providing a competitive edge and supporting the strategic decision making process within an organization. Knowledge can be represented in many forms, such as logical rules, decision trees, fuzzy rules, Bayesian belief networks, and artificial neural networks. The ways in which the knowledge can be formulated are called models (and sometimes classifiers or estimators). According to [1] a model can be defined as a description of a causal relationship between input and output variables.

However, data mining is an expensive activity and establishing a data mining infrastructure is a costly exercise for many small to medium scale organizations (expensive software, steep learning curve, need for specialized people). On the other hand, young organizations can benefit from the knowledge acquired in a completely different context. Therefore enabling knowledge to be shared and transferred between different systems can prove to be feasible. During the past several years, the Data Mining Group [2] has been working in this direction specifying the Predictive Model Mark-up Language or PMML [3], a standard XML-based language for interchanging knowledge between applications. PMML can be also used for knowledge integration of results obtained by mining distributed data sets.

DeVisa is a system based on collecting knowledge in one or more domains and further providing it as a service to consumers in the context of Semantic Web applications. It manages a repository of PMML models as a native XML database and exploits their predictive or descriptive nature through a specialized PMML query language named PMQL, being part of DeVisa. The current work develops several ideas described in [4] by clarifying the methods through which the models are processed. Also, in a previous work [5] a more restricted approach is presented, in which decision models are stored in a native XML database systems and processed via XQuery scripts.

I. PERSISTENCE AND SERIALIZATION OF PREDICTIVE MODELS

The PMML language is an XML-based approach for describing the predictive models that are generated as the output of data mining processes. It provides a declarative approach for defining self-describing data mining models. In consequence it has emerged as an interchange format for prediction models. It provides a clean interface between producers of models, such as statistical or data mining systems, and consumers of models, such as scoring systems or applications that employs embedded analytics. A PMML document can describe one or more models. According to the specifications, a PMML document consists of the following components:

- Data Dictionary. The data dictionary defines the fields that are the inputs to the models and specifies the type and value range for each field.
- Mining Schema. Each model contains one mining schema, which lists the fields used in the model. These fields are a subset of the fields in the Data Dictionary. The mining schema contains information that is specific to a certain model, while the data dictionary contains data definitions that do not vary with the model. For example, the Mining Schema specifies the usage type of an attribute, which may be active (an input of the model), predicted (an output of the model), or supplementary (holding descriptive information and ignored by the model).
- Transformation Dictionary. The Transformation Dictionary defines derived fields. Derived fields may be defined by normalization, which maps continuous or discrete values to numbers; by discretization, which maps continuous values to discrete values; by value mapping, which maps discrete values to discrete values; or by aggregation, which summarizes or collects groups of values, for example by computing averages.

Model Statistics. The Model Statistics component contains basic univariate statistics about the model, such as the minimum, maximum, mean, standard deviation, median, etc., of numerical attributes.

A method of persisting the PMML models in a database was depicted in [6], which presents a PMML-based scoring engine named Augustus. Augustus is an open source infrastructure for building and deploying data mining, and statistical models for large data sets and high volume data streams.

II. Dynamic Methods for Integrating Knowledge in Applications

In the context of the growing need for scalability and flexibility in the software systems, the SOA based architectures are increasingly adopted. The term Service Oriented Architecture [7] refers to the design of a distributed system. SOA is not a new technology. It is a novel design methodology and architecture aimed at maximizing the reuse of multiple services (possibly implemented on different platforms and using multiple programming languages). Some of the key principles of service-orientation that are specific to SOA paradigm are: loose coupling, autonomy, reusability, composability, abstraction from underlying logic and discoverability [8].

SOA is a concept for large heterogeneous distributed systems, in which loosely coupled pieces of software that encapsulate their logic within a distinct context and can be easily combined into a composite solution. It can be applied to knowledge capabilities as well. In the context of such a distributed application knowledge is provided as Web Services in one or more components.

In the rest of the section a few approaches in this direction are described.

In [9] it is presented an architecture in which a knowledge service provider, via its knowledge server, answers queries presented by some knowledge consumers. The knowledge server maintains a set of knowledge models and answers queries based on those knowledge models. For the consumers it may be expensive or impossible to obtain the answers otherwise.

In the context of automated knowledge elicitation from data sources, [10] proposes two models of organization of the data mining service providers and the interaction with their clients. The paper also describes a XML based language for data mining task requests and the functionality and services of data mining service providers.

Due to the increasing usage of the web space as a platform (one of the main goals of Web 2.0), there are many other proposals of data mining services available on the web such as [11].

[12] describes a web-based system for classifier sharing and fusion named CSF/DC. It enables the sharing of classification models, by allowing the up- and download of such models expressed in PMML on the system's online classifier repository. It also enables the online fusion of classification models located at distributed sites, which should be a priori registered in the system using a Web form. Unlike CSF/DC, DeVisa only supports processing of local models exposing the functionality as web services.

III. The Functional Architecture of DeVisa

From the functional perspective, DeVisa presents several services in interaction with two main client roles: data mining producer application and data mining consumer application (figure 1).

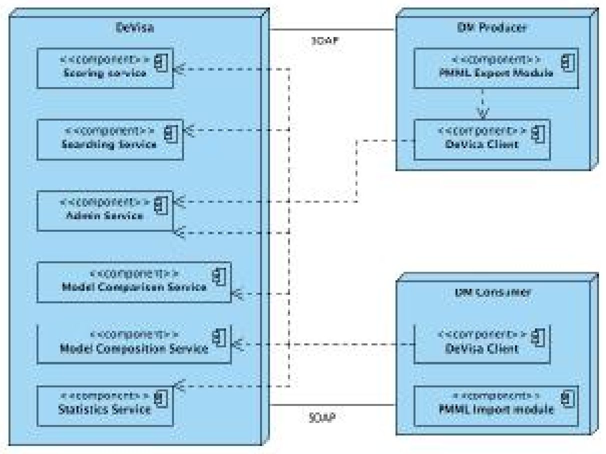
DeVisa
<<component>> Scoring service
<<component>> Searching Service
<<component>> Admin Service
<<component>> Model Comparison Service
<<component>> Model Composition Service
<<component>> Statistics Service
SOAP
DM Producer
<<component>> PMML Export Module
<<component>> DeVisa Client
DM Consumer
<<component>> DeVisa Client
<<component>> PMML Import module
SOAP

Figure 1. The DeVisa main functional blocks.

The DM (Data Mining) Consumer is a data mining application or a a Web Service client application that interacts with the PMML prediction models stored in DeVisa through a Web Service interface.

A DM Consumer also contains one of the following:

- A PMML import module: a module that transforms a PMML model into an internal model in order to further process it into decision or prediction processes (the DM consumer contains a scoring engine itself);
- A module that interprets a scoring result and integrates this result into its business logic (this is the case where the DM consumer does not include a scoring engine);

The DM Producer is a Web Service client application - typically a data mining application, like Weka- who uploads models in the DeVisa Repository. If the model that the DM producer has uploaded already exists in the DeVisa repository then and the model is newer than the existing one then it is replaced. It might need to be authenticated in order to execute the required function. For the purposes of testing and proving the concept we used a data mining system called Weka [13] as the main DM producer. At this moment Weka does not implement PMML support, therefore we created a light Weka PMML export module that typically maps the model classes to PMML documents.

The rest of the section gives an overview on the main functional services that DeVisa provides.

A. *Scoring Service*

The PMML Scoring Service is a module that offers scoring services via a SOAP interface.

The scoring use case means applying the models on the new instances. Depending on the models, there are several types of DeVisa scoring procedures:

- Classification Scoring. The scoring method receives as input a set of instances and one or more classification models and classifies the instances with respect to the models.
- Cluster Scoring. The scoring method receives as input a set of instances and one or more clustering models and assigns the instances to

the most appropriate cluster in each of the models.

- Association rules scoring. The scoring method receives as input a set of items (instances) and one or more association rule models. It determines all rules of each of the input models whose antecedent item set is a subset of a the input item set and returns the consequents of these rules as the inferred item sets. An extension of this procedure computes all rules whose antecedent and consequent item sets are included in the input item set. This version is useful to determine which item sets support which rules.

The model can be specified by its unique name that includes the producer's namespace, or can be selected by specifying desired characteristics related to freshness, performance measures, producer application, complexity etc.

There are three ways in which a client application -a DM consumer that invokes a scoring method- requests a model in DeVisa.

- *Exact Model.* the document refers a model in the DeVisa catalog; The DeVisa engine will use the specified model to execute the scoring task
- *Exact Schema.* the document refers a data dictionary but does not refer a model; It can specify desired properties though. DeVisa engine will select the appropriate model.
- *Match Schema.* the document describes a data dictionary (and possibly a mining schema according to the data dictionary). In this case the DeVisa engine is responsible for identifying the matching schema and the appropriate model (should these exist).

Except for the first case, the model is considered to be laxly specified.

B. Searching Service

DeVisa provides searching functions in the catalog as selecting the models with desired properties (e.g. performance measures, fields statistics, function type, degree of freshness etc.) or full text search in the model repository.

C. Model Comparison Service

Two schema compatible models can be compared from a qualitative perspective (e.g accuracy) and from a structural perspective (through XML differencing).

D. Model Composition Service

Model Composition allows the combination of simple models into a single composite PMML model. The PMML specification describes two types of composition:

Model sequencing is the process through which two or more models are combined into a sequence where the results of one model are used as input in another model. Model sequencing is supported partially by the PMML specification.

Examples of sequencing include:

- The missing values in a regression model can be replaced by a set of rules (or decision tree)
- Several classification models with the same target value can be merged via a voting scheme, i.e. the final classification result can be computed as an average of the results of the initial classifiers. The average can be computed by a regression model.
- Prediction results may have to be combined with a cost or profit matrix before a decision can be derived. A mailing campaign model may use tree classification to determine response probabilities per customer and channel. The cost matrix can be appended as a regression model that applies cost weighting factors to different channels, e.g., high cost for phone and low cost for email. The final decision is then based on the outcome of the regression model.

Model selection is the process of combining multiple 'embedded models', like model expressions, into the decision logic that selects one of the models depending on the current input values. For instance, a common method for optimizing prediction models is the combination of segmentation and regression. Data are grouped into segments and for each segment there may be different regression equations. If the segmentation can be expressed by decision rules then a decision tree can implement this kind of segment-based regression where any leaf node in the tree can contain an embedded regression model.

PMML version 3.2 supports the combination of decision trees and simple regression models. More general variants would be possible and may be defined in future versions of PMML.

In PMML Model composition uses three syntactical concepts:

- The essential elements of a predictive model are captured in elements that can be included in other models,
- Embedded models can define new fields, similar to derived fields and
- The leaf nodes in a decision tree can contain another predictive model.

In DeVisa simple models can be combined into more complex ones forming new valid PMML documents. A client application can choose among the existing models and the combination method. DeVisa identifies the specified models. If the models do not exist in the repository then the process stops. The found models are checked for compatibility. If they are not compatible the process stops. The new valid model is returned to the user or stored in the DeVisa repository.

E. Statistics Service

An application can invoke this service to obtain statistics on the models (e.g. frequencies per domain, schema, or function type etc.).

F. The Admin PMML Service

This service is based on XMLRPC or SOAP protocols and consists of methods for storing and retrieving PMML models. DeVisa redefines the basic SOAP store / retrieve web service with customized PMML features. Therefore, when a model is uploaded in the repository, it is validated against the PMML Schema, it is distributed in the appropriate collection (based on the domain / producer) and the catalog is updated with the new model's metadata. Furthermore the service provides features for updating/replacing an existing model with a newer one via XUpdate [14] or XQuery Update Facility [15] instructions, both supported by the underlying XML database system (eXist).

IV. THE BUILDING BLOCKS OF DEVISA

In order to achieve its main role as a knowledge provider through scoring services DeVisa comprises a modular technical architecture. Its main building blocks are: the metadata catalog, the repository, the PMQL language and the web services.

A. *The Metadata Catalog*

The Metadata Catalog component is a view on the PMML repository containing information on the PMML models in the repository. The Metadata Catalog is used in processing PMQL (see subsection V.C) queries in the annotation phase prior to actual query execution.

In DeVisa the Metadata Catalog is strongly dependent on the underlying XML database indexing system. The indexing system in eXist is based on pluggable indexing pipelines and is independent of the content (makes nearly no assumptions about the data to be indexed). The current version of eXist by default includes the following types of indexes: structural (the nodal structure, elements and attributes, of the documents in a collection), range (map specific text nodes and attributes of the documents in a collection to typed values), full text (map specific text nodes and attributes of the documents in a collection to text tokens), NGram (map specific text nodes and attributes of the documents in a collection to split tokens of n-characters), spatial (map elements of the documents in a collection containing geo-referenced geometries to dedicated data structures that allow efficient spatial queries).

Each entity represented in the DeVisa Metadata Catalog can specify imported domain ontologies as the terminology to define the concepts used in the models. These semantic annotations are used in the model discovery phase(VI-B)for resolving the ongoing request against the formal vocabulary used by the models. In case no ontology is specified, then the textual descriptions or keyword scan also contribute to a more accurate model discovery process.

DeVisa Metadata Catalog uses the indexing engine to capture the relationships that are depicted in Figure 2.

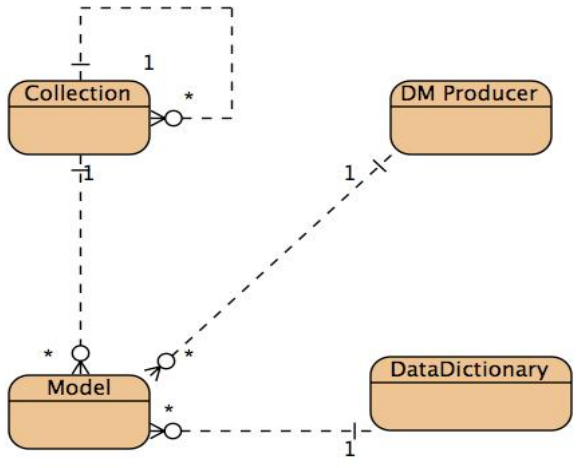

Figure 2. The Concepts and Relations that are modeled by the Metadata Catalog.

B. *The Model Repository*

The model repository uses the facilities offered by the underlying native XML database system. DeVisa uses eXist [16] for managing the PMML repository. It stores XML data according to the XML data model and features efficient, index-based XQuery processing. eXist supports many (web) technology standards making it an suitable application platform and is highly compliant with the XQuery standard. The query engine is extensible and features a large collection of XQuery Function Modules. eXist provides a suite of indexing types, such as range, full text. structura, spatial, Ngram.

C. *PMQL*

For the purpose of interacting with PMML documents DeVisa defines a language called PMQL (Predictive Model Query Language). PMQL is a query language based on the XML [17] syntax. A PMQL query is executed against the DeVisa repository of prediction models stored in PMML format. The PMQL schema and specifications can be found in [19]. A client application can wrap a PMQL query in a SOAP message and invoke a service method of the PMML Service..

Depending on the task, a PMQL query specifies both the target models it is based on, (exact models or through the schema, producer application, algorithm or other filters) and the instances to predict. PMQL can express queries that correspond to core DeVisa functionality (See Section IV).

In DeVisa a PMQL query is processed and executed by the PMQL Engine in the following phases:

- *Annotation*: syntactic checking (against PMQL Schema) and semantic checking (against metadata catalog, permission checking)
- *Rewriting* By solving the types and names mismatches and converting the query to an abstract tree. Types can be resolved by applying the allowed (or default) type conversions as specified in the XMLSchema. Name solving is subject to intense research in the context of achieving the Semantic Web desiderates. A promising approach is the use of ontologies to mediate between heterogeneous schemas. In this way a PMQL query can be transformed and interpreted with respect to the internal schema of a certain PMML model in the DeVisa repository.
- *Optimization.* This phase makes use of the query optimization facilities specific to the underlying XML database system. It analyzes the query at compile time and searches for optimizable sub-expressions within the query tree. If it finds optimizable expressions, the optimizer will modify the query and wrap some special instructions around the optimizable code block.
- *Plan building.* The query plan is a pair (mode, instances) contains a clear reference to the models subject to scoring and a clear reference to an XML document containing the instances with the schema reinterpreted with respect to the aforementioned model. In the case there are several models that satisfy the query requirements the query plan is made of a sequence of execution units.

- *Execution.* This phase actually invokes the stored XQuery functions that process the PMML repository and computes the answer to the query. Expressing the scoring algorithms on the PMML models in XQuery leverages the underlying XML database query optimization.

D. *Web Service*

The DeVisa system provides its functionality via web services definitions. The Web Services technology provides an efficient way to share application logic across multiple machines running various operating systems and using different development environments. To achieve this, Web Services utilizes the SOAP [19], WSDL [20], XML Schema [21] and some other XML-based technologies, providing a standards-based approach to overcoming the platform and language differences [8].

The DeVisa Admin PMML Service supports SOAP, XMLRPC and REST-style protocols. The PMML Model Service supports SOAP and REST-style messaging. The web services are described using WSDL, are written in Java and deployed using Apache Axis 2 framework [22].

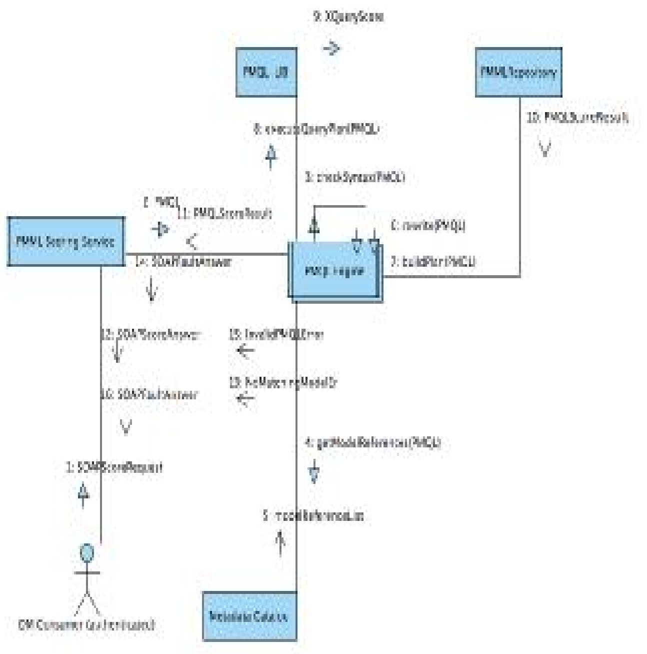

Figure 3. The Scoring Service. The communication diagram among DeVisa Components.

V. FOCUS ON THE SCORING SERVICE

We consider now the scoring use case in DeVisa, which is depicted in figure 3.

The scoring procedure takes place in the subsequent phases. The DM Consumer sends a SOAP message that wraps the request for a scoring task expressed in PMQL.

The PMQL scoring query is transmitted to the PMQL engine, which is responsible with processing and resolving PMQL requests. PMQL Engine executes the PMQL processing phases described in subsection V.C.

If the PMQL coming from the DM Consumer is not valid (does not match the PMQL Schema) then an “Invalid PMQL” error is returned, which is wrapped in a SOAP error message.

Then the engine checks in the repository containing the requested models. In the case in which there is no model in the repository that satisfies the request, then a "No Matching Model Error" is returned to the PMML Service and subsequently to the consumer.

In the successful case the PMQL Engine rewrites the PMQL scoring query so that it contains exact references to the models to score on.

In the plan building phase the engine transforms the PMQLDocument into an array of PMQLDocument elements so that each matching model is paired with the set of instances. Each of the elements needs to be transferred to an XQuery function and executed against the PMML repository.

The answer to a scoring request is a PMQL document, containing the set of instances in the request together with the score for each of them. The score can represent for instance the value of a classification class attribute or a cluster id that represents the assignment of the instance to the specified cluster.

For instance if we consider a loan module in a banking application, DeVisa can assist the decision of giving a loan to a customer. In the repository DeVisa stores and maintains models resulted from the previous loans records. Based on the characteristics of the new customer’s profile the DeVisa system can provide a score concerning customer’s reliability that helps the loan officer in taking the decision.

Because DeVisa is an stand-alone system that communicates with other applications via web services, it can be integrated using the SOA paradigm (see section III) in legacy applications as well as in new modular applications.

As described in [24] the possibility to integrate DeVisa as a decision support module into an e-Health system called Telemon based on SOA architecture is currently investigated.

VI. CONCLUSION

In this paper an online system for scoring and management (DeVisa) of prediction models was described. Its flexible architecture allows dynamic integration into distributed applications as a knowledge provider component. The system has been designed to provide unified access to different prediction models using standard technologies based on XML, namely PMML. The repository layer uses a native XML database management system for managing the PMML models.

The system provides functions such as scoring, model comparison, model composition, searching, statistics or administration through a web service interface. DeVisa also defines a specialized PMML query language named PMQL used for specifying client requests and interaction with PMML repository. The paper focuses on the most important function of the system, namely the scoring function, describing main phases.

In the future work DeVisa will include a mash-up interface for the scoring and composition services. The system will be able to recommend the appropriate service by selecting the appropriate model. Furthermore DeVisa will create a bridge between Web Service composition and model composition.

REFERENCES

[1] K. Cios, W. Pedrycz, R. Swiniarski, and L. Kurgan, Data Mining. A Knowledge Discovery Approach. Springer, 2007.

[2] Data mining group. Http://www.dmg.org. [Online]. Available: http://www.dmg.org, 2008, January

[3] Pmml version 3.2. Http://www.dmg.org/pmml-v3- 2.html. [Online]. Available: http://www.dmg.org/pmml- v3- 2.html, October, 2007

[4] D. Gorea, "Towards storing and interchanging data mining models," in Proceedings of the 3rd Balkan Conference in Informatics, vol. 2, 09 2007, pp. 229–236. [Online]. Available: http://bci2007.uni- sofia.bg

[5] D. Gorea and S. Buraga, "Towards integrating decision tree with xml technologies," in Proceedings of the 8th International Conference on Development and Application Systems – DAS 2006, Suceava University Press. Suceava University Press, 2007.

[6] J. Chaves, C. Curry, R. L. Grossman, D. Locke, and S. Vejcik, "Augustus: the design and architecture of a pmml-based scoring engine," in DMSSP '06: Proceedings of the 4th international workshop on Data mining standards, services and platforms. New York, NY, USA: ACM, pp. 38 – 46, 2006.

[7] T. Erl, Service-Oriented Architecture: Concepts, Technology, and Design. Prentice Hall, 2005.

[8] Y. Vasiliev, SOA and WS-BPEL. Packt Publishing, 2007.

[9] S. Xu and W. Zhang, "Knowledge as a service and knowledge breaching," in SCC '05: Proceedings of the 2005 IEEE International Conference on Services Computing. Washington, DC, USA: IEEE Computer Society, pp. 87–94, , 2005.

[10] S. Krishnaswamy, S. W. Loke, and A. Zaslavsky, "Knowledge elicitation through Web-based data mining services," Lecture Notes in Computer Science, vol. 2176, pp. 120, 2001

[11] M.-H. T. Chieh-Yuan Tsai, "A dynamic web service based data mining process system," in The Fifth International Conference on Computer and Information Technology CIT 2005, pp. 1033– 1039, 2005.

[12] I. V. Grigorios Tsoumakas, "An interoperable and scalable web-based system for classifier sharing and fusion," Expert Systems with Applications, vol. 33, no. 3, pp. 716–724, October 2007.

[13] Weka 3 - data mining software in java. Http://www.cs.waikato.ac.nz/ml/weka. [Online]. Available: http: //www.cs.waikato.ac.nz/ml/weka 2007.

[14] Xupdate - xml update language. [Online]. Available: http: //xmldb-org.sourceforge.net/xupdate/ 2003.

[15] Xquery update facility 1.0. [Online]. Available: http://www.w3.org/TR/xquery- update- 10/, 2007, August

[16] Exist - open source native xml database. Http://www.exist- db.org/. [Online]. Available: http://www.exist- db.org/ 2007.

[17] Extensible markup language (xml) 1.0 (fourth edition) w3c recommendation. [Online]. Available: http://www.w3.org/TR/XML/ 2006.

[18] Devisa. Http://devisa.sourceforge.net. [Online]. Available: http: //devisa.sourceforge.net, 2007.

[19] Soap specifications. [Online]. Available: http://www.w3.org/TR/ soap/, 2007.

[20] E. Christensen, F. Curbera, G. Meredith, and S. Weerawarana.

[21] Wsdl - web services description language. [Online]. Available: http://www.w3.org/TR/wsdl, 2001.

[22] C. M. Sperberg-McQueen and H. Thompson. Xml schema. [Online]. Available: http://www.w3.org/XML/Schema, 2000.

[23] Apache axis. Http://ws.apache.org/axis/. [Online]. Available: http://ws.apache.org/axis/, 2007.

[24] L. Alboaie, D. Gorea, V. Felea, "Semantic Integrity Control in the Database Layer of an e-Health System" ,International Conference on Computers, Communications and Control, ICCCC 2008, Baile Felix, Romania, 2008.