EbXML Registry/Repository Implementation

Ivan Magdalenić, Ivan Matasić, Damir Pintar, Ivo Pejaković, Mihaela Sokić
Faculty of Electrical Engineering and Computing, University of Zagreb
Unska 3, HR-10000 Zagreb, Croatia
Tel: +385 1 6129 756; Fax: +385 1 6129 616
E-mails: ivan.magdalenic@fer.hr, ivan.matasic@fer.hr, damir.pintar@fer.hr, ivo.pejakovic@fer.hr, mihaela.sokic@fer.hr

Abstract: **There are several B2B frameworks at the market, ebXML being one of them. Market position of ebXML is getting stronger due to its interoperability and strong support by UN/CEFACT and OASIS. EbXML has modular architecture consisting of five layers: Core Components, Business Process, Trading Partner Agreement, Registry/Repository and T,R,P (Transport, Routing and Packaging). Modularity of ebXML specifications enables implementation of layers without the need to implement all of its functionality. This paper describes implementation model of Registry/Repository layer. Role of Registry/Repository in ebXML is vital as it stores all necessary information to perform B2B transactions and gives a standardized methodology for retrieval and storage of such information in distributed environment.**

1. INTRODUCTION

Large number of companies is turning into new economy enabled with the evolution of digital media. Several standards for B2B (Business to Business) communication have been pushed into market by leading software companies, supported by big industrial companies or branched consortiums.

The problem space in such kind of implementation of different "private" B2B models is obviously in interoperability. Partner companies need to implement B2B models established by industry leaders, so they are often forced to change their own business models to conduct B2B transactions. Complexity in implementing different business models to comply with business models for each new partner discourages companies, especially small and medium sized ones, to transfer to B2B trading. Development costs, as result of absence of standardized B2B model, are also a huge problem.

B2B model has to be flexible, expandable, interoperable, and above all standardized. EbXML (Electronic Business using eXtensible Markup Language) is already perceived as one of greatest global standards for B2B communication. EbXML as a specification is a solution to a variety of problems in business communication between trading partners that emerge as a result of current systems incompatibility. Section 2 of this paper gives ebXML basics, insight of Registry/Repository layer and its role. Section 3 describes our implementation of Registry Information Model with the description how new technologies such as Enterprise Java Beans are used. Conclusion is given in section 4, followed by a list of references.

2. EbXML BASICS

EbXML sponsored by UN/CEFACT and OASIS, is a modular suite of specifications that enables enterprises of any size and in any geographical location to conduct business over the Internet. Using ebXML, companies now have a standard method to exchange business messages, conduct trading relationships, communicate data in common terms and define and register business processes [1].

Figure 1. Presents technological view on ebXML standard.

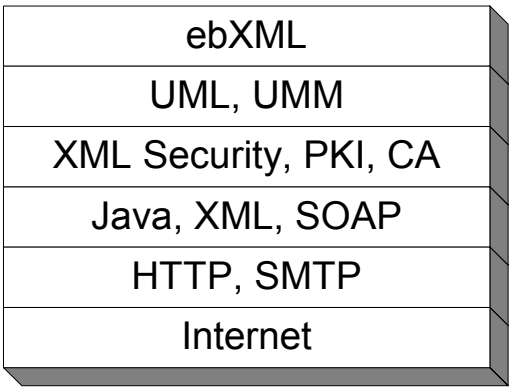

Figure 1. EbXML – technological view

As it can be seen, large number of technologies based on Internet, provides solution to B2B transactions in ebXML. XML (eXtensible Markup Language) [2], [3] is used for document standardization and description of information through ebXML. A eBTL (electronic Business Technology Laboratory) [4] established at Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia is conducting series of experimental projects regarding ebXML and security solutions.

As a global framework for e-business data exchange, ebXML [1] consists of five architectural components: TRP (Transport, Routing and Packaging), Registry/Repository, TP (Trading Partner), BP (Business Process) and CC (Core Components). All ebXML components cooperate in fulfillment of ebXML. For example, EbXML TRP uses information's defined in ebXML TP [5], while ebXML TP can be stored through ebXML Registry [6]. EbXML BP is implemented through ebXML TRP, TP and Registry. All components have elements of transport and content security implemented. We need to mention that, compared to others, Core Components are still not completely defined part of ebXML standard.

2.1. EbXML Registry/Repository

EbXML Registry can be described as ebXML brain, as it stores all information that ebXML trading partners need (e.g. information's on trading partners and their CPP – Collaboration Protocol Profile). Figure 2 describes ebXML Registry/Repository architecture and its reference to other objects and business or Internet services.

From the Figure 2 it is obvious that term Repository describes mechanism for data storage, while Registry describes interface services through which Repository objects can be reached. Data in Repository is viewed, inserted, stored and deleted through user requests on Registry. Repository and Registry Services are separated so that one can use repositories from multiple different clients through only one Registry Interface.

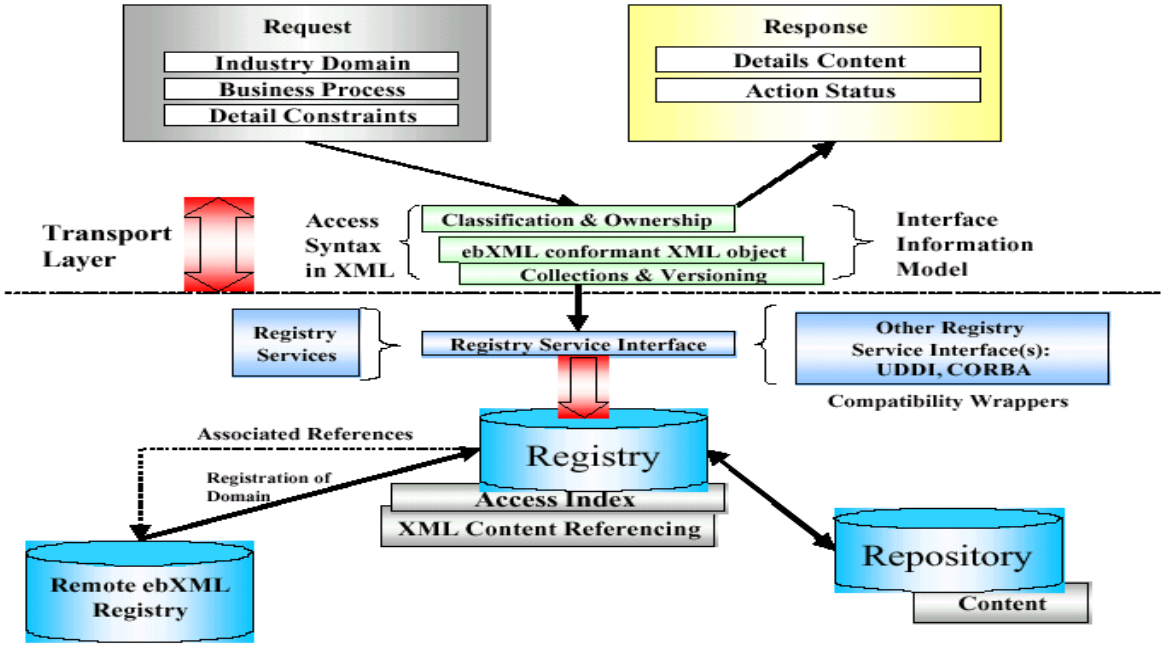

Figure 2. EbXML Registry/Repository

Two specifications define Registry/Repository Service:

- *Registry Information Model* (RIM) describing objects that can be stored in Repository, meta data on those objects, and structure of Repository;
- *Registry Service Specification* (RS) describing detailed view on interfaces for users, as well as functionality of Registry Services in those interfaces.

High-level public view of Registry Information Model is given in Figure 3, in form of a UML Class diagram. In this case UML diagrams are used as a very powerful way to concisely describe concepts. The Registry Information Model provides a blueprint or high-level schema for the ebXML *Registry*. Its primary value is for implementers of ebXML *Registries*. It provides these implementers with information on the type of metadata that is stored in the *Registry* as well as the relationships among metadata *Classes*.

The ebXML Registry architecture consists of an ebXML Registry and ebXML Registry Clients. The Registry Client interfaces may be local to the Registry or local to the user. There are three possible topologies for Registry Client:

- Registry can provide a Web based "thin client" application for accessing the Registry that is available to the user using a common Web browser. In this scenario the Registry Client interfaces reside across the Internet and are local to the Registry from the user's view.
- The user can also use a "fat client" Registry Browser application to access the Registry. In this scenario the

Registry Client interfaces reside within the Registry Browser tool and are local to the Registry from the user's view. The Registry Client interfaces communicate with the Registry over the Internet in this scenario.

– A third topology made possible by the Registry architecture is where the Registry Client interfaces reside in a server side business component such as a Purchasing business component. In this topology there may be no direct user interface or user intervention involved. Instead the Purchasing business component may access the Registry in an automated manner to select possible sellers or service providers based on current business needs.

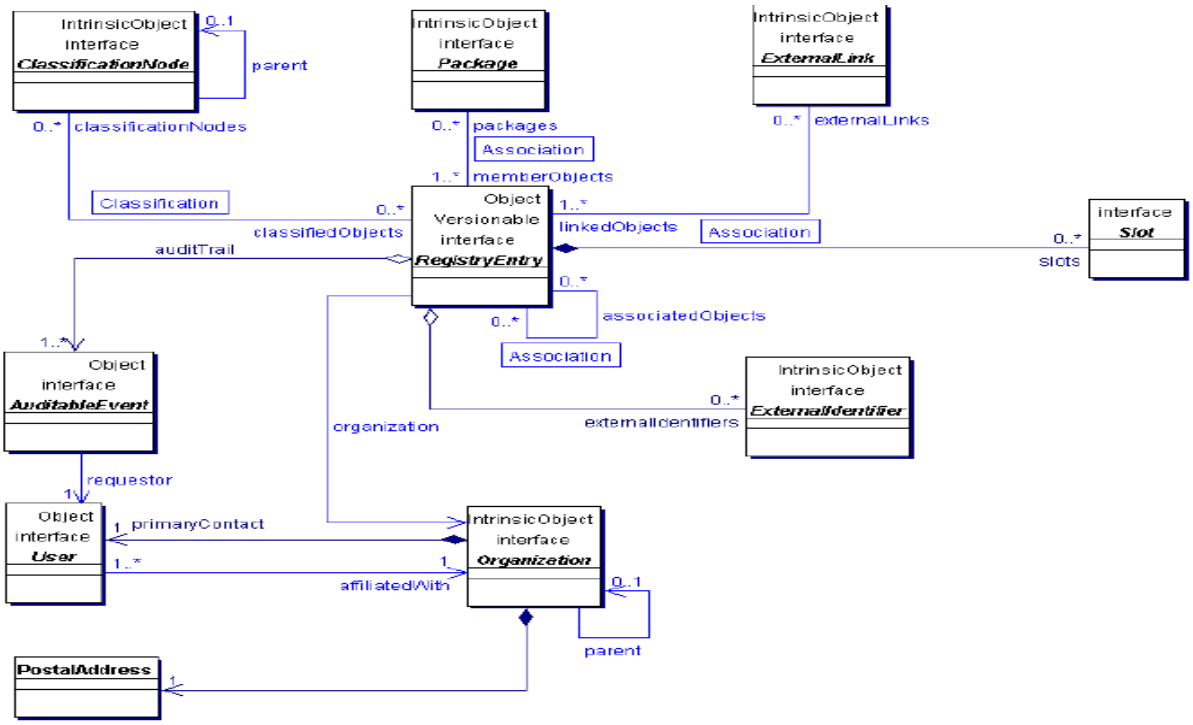

Figure 3. Registry Information Model (High level public view) in form of a UML Class diagram

3. IMPLEMENTATION OF EbXML RIM

Our implementation of Registry Information Model conforms to specification of the Information Model for the ebXML *Registry* version 2.0 [7]. The document defining Information Model for the ebXML *Registry* is part of developing specifications for interoperable XML registries and repositories and is an approved Committee Specification of the OASIS ebXML Registry Technical Committee. Current status of this document is a DRAFT Specification.

The Registry provides a stable store where information submitted by *a Submitting Organization* (organization that provides data to the Registry) is made persistent. Such information is used to facilitate ebXML-based B2B partnerships and transactions.

Submitted content includes but is not limited to XML schema and documents, process descriptions, ebXML Core Components, context descriptions, UML models, information about parties and even software components.

Metadata, used to describe the stored content, is defined in the Registry Information Model. The Registry Information Model defines what types of objects are stored and how stored objects are organized in the *Registry*. It does not deal with the actual content of the Repository. All *Elements* of the information model represent metadata about the content and not the content itself.

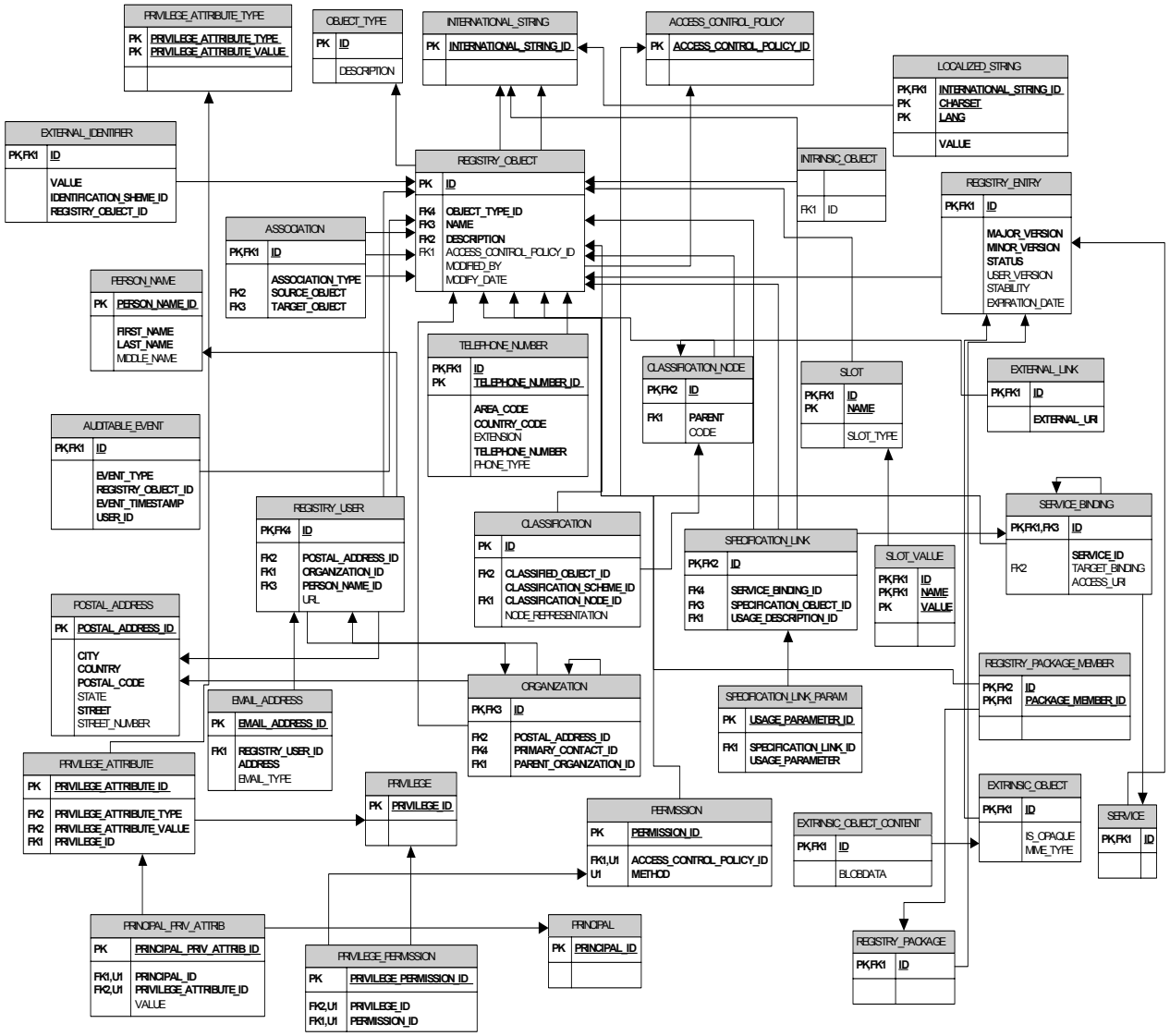

Figure 4: Tables in ebXML Repository

The Registry Information Model has been implemented within an ebXML Registry in the form of a relational database. Relation database model has been developed using standard methodology (E-R modeling) and Oracle Designer 6i tool. Classes specified in Registry Information Model have been mapped to relation database tables. Inheritance between classes in the information model has been implemented as relationships in database.

Database tier, founded on Oracle 8i technology, is used for storing all submitted content and platform related data. Figure 4 presents database tables and relations between database tables as connections, with primary and foreign keys as special marks. Although our implementation is located on commercial database, our solution doesn't use any of product specific features provided by RDBMS. Thus, our solution can be implemented using any of relation databases that conform to ANSI SQL92 standard. Various rules assigned to specific objects and attributes in information model that cannot be implemented in data model without use of product specific features are implemented in application layer in order to provide product-independency of database layer.

EJB (Enterprise Java Beans) is relatively new server-side component model developed within part of J2EE platform. EJB infrastructure includes EJB classes that represent beans,

application server that manages beans and database that ensures persistency.

We found EJB to be very suitable for implementation of RIM for several reasons:

- EJB technology is specially designed for scalable distributed applications,
- EJB considers transaction management, security mechanisms and data persistency as part of its application server - these features are therefore automatically managed.

Java environment ensures that software solution will be open and applicable through different platforms. During the starting implementation phases we decided to use relatively new EJB 2.0 specification instead of EJB 1.1 specification used in development of previous applications. Reasons for such a decision were:

- New EJB 2.0 specification is not compatible with older version and it can be assumed that future application servers will be optimized for EJB 2.0 version (old EJB 1.1. version will be supported only for compatibility reasons);
- EJB version 2.0 brings a new way of automatic management of persistency, which enables faster development time and ensures openness of our solution;
- One of new EJB features is automatic management of references between entities, and this feature showed to be very useful during development;
- A new kind of EJB beans, *message-driven* beans, serves for asynchronous message transactions, gives a new dimension to flexibility and modularity in light of developing specifications for ebXML messaging.

We have selected BEA Weblogic Server for application server as it is free (for developers), but also because it is very well documented and globally accepted. All classes defined in RIM are represented through entity beans whose persistency is managed by Weblogic application server. RS will be designed through session beans that will manage entity beans in serving different demands from outside Registry/Repository.

A set of *Registry Services* that provide access to *Registry* content to clients of the *Registry* is defined in the ebXML Registry Services Specification [8] and in this part of description has not been taken into consideration.

4. CONCLUSION

EbXML is a specification for B2B e-commerce. EbXML Registry can be described as ebXML brain, as it stores all information that ebXML trading partners need.

The Registry Information Model (ebXML RIM) provides a blueprint or high-level schema for the ebXML *Registry*. Its primary value is for implementers of ebXML *Registries*.

To create modular and flexible implementation of ebXML RIM we have decided to use EJB v.2.0, Weblogic application container and Oracle 8i database. We found that many features provided by these technologies (especially EJB) were very useful during implementation of RIM and very helpful during solving many of the perceived problems.

We found these technologies (especially EJB) to be able to implement RIM and many of the perceived problems with their features. Comparing this solution to several others developed (or in development) in the world, we found it to be more scalable and offer great expandability, which is crucial in implementing future versions of specifications and therefore adding new features. As there are only few open-source or published Registry implementations in the world (even from software vendors) we could not establish clear vision on Registry/Repository implementation that would be predominant in the future. We think that as long as the specifications are strictly obeyed, different methodology in constructing database and other layers will not stand in the way of functionality and interoperability of our artifact.

We consider our implementation as a preparation of implementation of RS, which should be done by the end of 2002. With Registry/Repository layer implemented, we can start achieving our primary goal: implementation of a real B2B communication between trading partners based on ebXML specifications.

REFERENCES

[1] www.ebXML.org

[2] www.xml.org

[3] www.w3.org

[4] Marko Topolnik, Damir Pintar, Mihaela Sokić: "Experimental Implementation of Emerging e-Business Technologies: EbXML and PKI", Proceedings of the MIPRO2002 (Electronic Commerce), Opatija, May 2002, p.p. 1-6.

[5] Ivan Magdalenić, Ivo Pejaković, Dražen Pranjić: "Business documents presentation and interchange using SOAP", Proceedings of the MIPRO2002 (Electronic Commerce), Opatija, May 2002, p.p. 13-17.

[6] Mihaela Vrhoci, Dubravka Đelmiš, Jarmila Maksimović: "Collaboration Protocol Profile (CPP) and Collaboration-Protocol Agreement (CPA) in B2B transactions", Proceedings of the MIPRO2002 (Electronic Commerce), Opatija, May 2002, p.p. 45-50.

[7] UN/CEFACT-OASIS: "EbXML Registry Information Model, v2.0", December 2001.

[8] UN/CEFACT-OASIS: "EbXML Registry Services Specification, v2.0", December 2001.