ISSN:0975-9646

Enhancing the Semantic of the Business Repository Components for More Efficient Web Services Discovery

Salim BAROUDI[#1], Mohamed BAHAJ[#2]

[#] *Department of computer science, Faculty of Science and Technology*
University Hassan I, Settat, Morocco

***Abstract* — In our days, ebXML and Web Services are two modern technologies used to perform collaboration and interactions between businesses in B2B contexts. In this paper, we will base on our last work presented in the article "Integrating Ontologies into ebXML Registries for Efficient Service Discovery" [1] to expose a complement approach to enhance the interpretation of the semantic by the Business Registry component called the QueryManager. This approach consist on storing the resulting semantic OWL constructs into the Business Repository and designing pre-stored functions for better interaction between the QueryManager and the Business Repository Components.**

Keywords* — *EbXML, Web Services, ebXML registries, semantic, Web Ontology Language (OWL), Business Registry, Business Repository.

I. INTRODUCTION

Electronic Business XML (ebXML) [2] is a standard from OASIS [3] and United Nations Center for Trade Facilitation and Electronic Business, UN/CEFACT that is technology who provides an infrastructure permitting to the enterprises to find each other's services, products, business processes, and documents in a standard way and thus helps to facilitate conducting electronic business.

ebXML 'Registry' and 'Repository' are two different components used to store semantic - meta-data and data- for the service. The registry meta-data are stored in the 'Registry', and the documents pointed at by the registry objects reside in a 'Repository'.

In this article that is an extension to our last work named "Integrating Ontologies into ebXML Registries for Efficient Service Discovery", we will present an efficient way to store the semantic OWL constructs into the ebXML repository and how we can exploit some pre-designed RDB procedures that we store in the same component to enhance the semantic interpretation by the Business Registry QuerManager component for an efficient Web service discovery.

To summarize our goal we will propose an efficient way to improve the QueryManager component and it interaction with the data of the Business Registry, thing that will guarantee an intelligent use of the whole service.

The two major approaches used for Semantic web Services based on Business Registries are OWL-S and Using OWL to introduce the semantic in the business registries. A lake of works and studies treat and propose process of making Web Services semantic. OWL-S [14] is s description language based on XML using the model of description logics that allows automatic discovery of web services using agents, automatic invocation of web services. The aim of the second approach –presented in article [1]- is introducing ontologies into business registries to make them an OWL aware.

II. EBXML

EbXML facilitates electronic business as follows:

In order for enterprises to conduct electronic business with each other, they must first discover each other and the products and services they have to offer. ebXML provides a registry where such information can be published and discovered,

An enterprise needs to determine which business processes and documents are necessary to communicate with a potential partner. A Business Process Specification Schema (BPSS) in ebXML provides the definition of an XML document that describes how an organization conducts its business,

After this phase, the enterprises need to determine how to exchange information. The Collaboration Protocol Agreement (CPA) specifies the details of how two organizations have agreed to conduct electronic business.

ebXML is often used in a Web Services collaboration context and especially to do the discovery and manage interactions between Web Services and clients. It used in many critical sectors like Bank and Finance, Agriculture, Construction, Medicine ….

A. ebXML registry architecture and information model

ebXML registry [3] provides a persistent store for registry content. It's consists of both a registry and a repository. The repository is capable to store any type of electronic content, while the registry is capable of storing meta-data that describes content. The content within the repository is referred to as 'repository items' while the meta-data within the registry is referred to as 'registry objects'. The repository store registry data in relational databases. ebXML Registry Services Specification defines a set of Registry Service interfaces which provide access to registry content. There are a set of methods that must be supported by each interface.

The –Fig. 1- presents all component and protocols that can be involved in interaction between clients and Web Services. A registry client program utilizes the services of the registry by invoking methods on one of these interfaces.

The Query Manager component also uses these methods to construct the objects by obtaining the required data from the relational database through SQL queries. In other words, when a client submits a request to the registry, registry objects are made by retrieving the related information from the database through SQL queries and are served to the user through the methods of these objects.

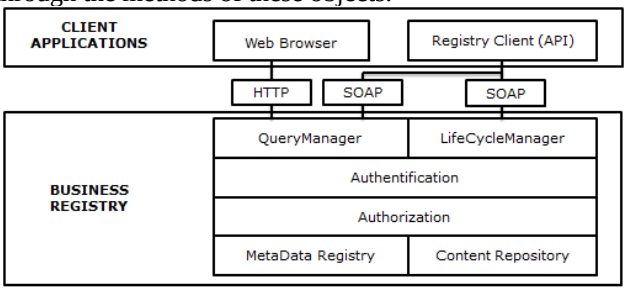

Fig. 1 View of ebXML Registry Architecture

An ebXML registry allows us to define semantics basically through two mechanisms: first, it allows properties of registry objects to be defined through 'slots' and, secondly, meta-data can be stored in the registry through a 'classification' mechanism. This information can then be used to discover the registry objects by exploiting the ebXML query mechanisms.

We note that within the scope of this paper, we address how to bring semantic to the ebXML registry using the Ontology concept.

B. ebXML registry information model (RIM)

ebXML registry information model. The ebXML registry defines a Registry In formation Model (RIM) which specifies the standard meta-data that may be submitted to the registry. This complements the ebXML Registry API which defines the interface clients may use to interact with the registry. The -Figure 3- presents the part of the ebXML RIM related with storing meta-data information.

III. ONTOLOGY AND WEB ONTOLOGY LANGUAGE (OWL)

A. Ontology

Ontology is a description of a content and relationship expressed as a formal vocabulary. This can be used to classify domain specific information in different areas such as in medicine, education, engineering ... etc.

Most of the business information stored in repositories in Business to Business (B2B) infrastructure frameworks is difficult to extract due to unstructured domain classifications in business registries [4].

Therefore use of ontology as an information source to formalize domain knowledge and finding a way of mapping this information to registry content is a practical solution to enhance B2B communication. We have used OWL ontology language to represent the web semantic content in the repository.

B. Web Ontology Language (OWL)

Web Ontology Language (OWL) [5] is a semantic markup language for publishing and sharing ontologies on the World Wide Web. OWL is derived from the DAML+OIL Web Ontology Language [6,7], and builds upon the Resource Description Framework (RDF) [8,9]. It uses Description Logic [10] as a main construct to specify information related to a specific domain.

OWL describes the structure of a domain in terms of classes, data type and object properties. Classes can be names (URIs) or expressions and the following set of constructors are provided for building class expressions: owl:intersectionOf, owl:unionOf, owl:complementOf, owl:one Of, owl:allValuesFrom, owl:someValuesFrom, andowl:hasValue -Figure 22-. Properties can have multiple domains and multiple ranges. Multiple domain (range) expressions restrict the domain (range) of a property to the intersection of the class expressions.

Another aspect of the language is the axioms supported. These axioms are specials properties that make possible to assert subsumption or equivalence with respect to classes or properties. The following are the set of OWL axioms : rdfs:subClassOf, owl:sameClassAs,rdfs:subPropertyOf, owl:samePropertyAs, owl:disjointWith, owl:sameIndividualAs, owl:differentIndividualFrom, owl:inverseOf, owl:transitiveProperty, owl:functionalProperty, owl:inverseFunctionalProperty – Fig. 2-.

RDF Schema Features	**(In)Equality**	**Property Characteristics**
– *Class (Thing, Nothing)* – *rdfs: subClassOf* – *rdf:Property* – *rdfs: subPropertyOf* – *rdfs:domain* – *rdfs:range* – *individual*	– *equivalentClass* – *equivalentProperty* – *sameAs* – *differentFrom* – *AllDifferent* – *distinctMembers*	– *ObjectProperty* – *DatatypeProperty* – *inverseOf* – *TransitiveProperty* – *SymmetricProperty* – *FunctionalProperty* – *InverseFunctionalProperty*
Property Restrictions – *Restriction* – *onProperty* – *allValuesFrom* – *someValuesFrom*	**Restricted Cardinality** – *minCardinality* – *maxCardinality* – *cardinality*	**Datatypes** –*xsd datatypes* **Class Intersection** –*intersectionOf*

Fig. 1 : Example of OWL Constructs and Properties

According to the complexity of the business domain, OWL is subdivided to three sub-languages: OWL Lite, OWL DL, and OWL Full.

OWL Lite: It's the elementary level of OWL. It's based on a low hierarchy class level.

OWL DL: It's more expressive than the OWL Lite, and support a maximum of expressiveness, decidability and completeness. Based on Description Logic defined in section 3.3.

OWL Full: Support a maximum of expressiveness and freedom is given to users to define their own RDF formats.

IV. RELATIONAL DATABASE

A relational database [11] is database that stores information about both the data and how it is related. "In relational structuring, all data and relationships are represented in flat, two-dimensional table called a relation." For example, organizations often want to store and retrieve information about people, where they are located and how to contact them. Often many people live or work at a variety of addresses. So, recording and retrieving them becomes important—relational databases are good for supporting these kinds of applications.

Each database is a collection of related tables. Each table is a physical representation of an entity or object that is in a tabular format consisting of columns and rows. Columns are the fields of a record or the attributes of an entity. The rows contain the values or data instances; these are also called records or tuples.

A. Primary key

A primary key uniquely specifies a tuple within a table. In order for an attribute to be a good primary key it must not repeat. While natural attributes (attributes used to describe the data being entered) are sometimes good primary keys, surrogate keys are often used instead. A surrogate key is an artificial attribute assigned to an object which uniquely identifies it (for instance, in a table of information about students at a school they might all be assigned a student ID in order to differentiate them). The surrogate key has no intrinsic (inherent) meaning, but rather is useful through its ability to uniquely identify a tuple. Another common occurrence, especially in regard to N:M cardinality is the composite key. A composite key is a key made up of two or more attributes within a table that (together) uniquely identify a record. (For example, in a database relating students, teachers, and classes. Classes could be uniquely identified by a composite key of their room number and time slot, since no other class could have exactly the same combination of attributes. In fact, use of a composite key such as this can be a form of data verification, albeit a weak one.

B. Foreign key

A foreign key is a field in a relational table that matches the primary key column of another table. The foreign key can be used to cross-reference tables. Foreign keys need not have unique values in the referencing relation. Foreign keys effectively use the values of attributes in the referenced relation to restrict the domain of one or more attributes in the referencing relation. A foreign key could be described formally as: "For all tuples in the referencing relation projected over the referencing attributes, there must exist a tuple in the referenced relation projected over those same attributes such that the values in each of the referencing attributes match the corresponding values in the referenced attributes."

C. Stored procedures

A stored procedure is executable code that is associated with, and generally stored in, the database. Stored procedures usually collect and customize common operations, like inserting a tuple into a relation, gathering statistical information about usage patterns, or encapsulating complex business logic and calculations. Frequently they are used as an application programming interface (API) for security or simplicity. Implementations of stored procedures on SQL RDBMSs often allow developers to take advantage of procedural extensions (often vendor-specific) to the standard declarative SQL syntax. Stored procedures are not part of the relational database model, but all commercial implementations include them.

V. PROPOSED APPROACH TO ENHANCE THE QUERYMANAGER SEMANTIC INTERPRETATION AND STORING SEMANTIC META-DATA INTO THE BUSINESS REPOSITORY

The goal of this article is to enhance the semantic of the ebXML repository –Fig. 3- by storing the meta-data (ebXML sematic constructs obtained by applying the OWL mapping approach presented in the article [1]) and designing pre-stored SQL procedures into this repository permitting the interpretation of the stored semantic by the business registry component named QueryManager.

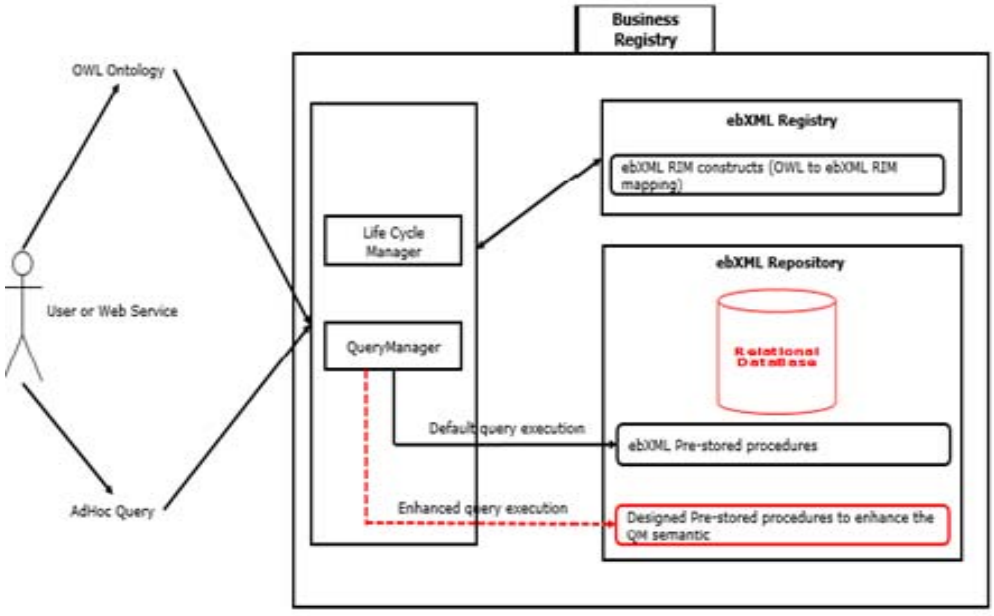

Fig. 2 : The enhancing approach

A. ebXML RIM specifications constructs storage into the Business Repository

In this step, we will present the chosen approach for storing the ebXML sematic constructs obtained by applying the OWL mapping approach presented in the article [1], into the ebXML repository relational Database.

The target Database schema presented in the –Fig. 4- is composed of three tables: Name, Association and ClassificationNode. Moreover, we not that to ensure the uniqueness of the Database entries, we propose to use not just the names of the ClassificationNodes and Associations, but the whole URI of the resource added before these names.

Table Name will contain the information about all the ClassificationNodes, ClassificationSchemes, Associations and the heritage between ClassificationNodes and Associations. (Examples in – Fig. 5, Fig. 6, Fig. 7 and Fig. 8-).

The –Fig. 6- is a screenshot of lines of the Mysql Database that represent the storage of the 'Goods' ClassificationsScheme.

Table Association will store all the Associations objects and the access security information to each Association. The AssociationType and the ObjectType entries will depend on the type and the object of the association, and will permit the representation of all types of OWL DL properties like transitiveProperties, sameAsProperties … .

In the example presented in –Fig. 9-, the storage of the Association title that is a DataTypePropoerty, is proceeded by specifying the objectType as a DataType and the associationTypePropoerty's sourceType as the anonymous class source of Association and targetType as the dataType of the association.

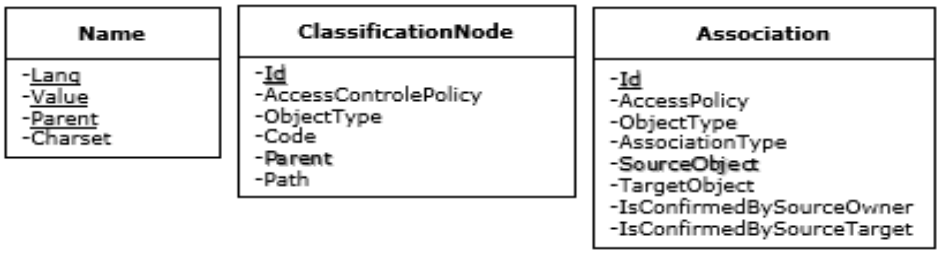

Fig. 4: ebXML Repository Relational Database

```
<ClassificationScheme id="Goods">
    <ClassificationNode id="Book" parent="Goods" code="book" />
    <ClassificationNode id="CD" parent="Goods" code="cd" />
</ClassificationScheme>
```

Fig. 5: ebXML RIM Classification hierarchies mapped construct example

lang	value	parent	charset
NONE	http://onto.ui.sav.sk/agents.owl#Goods	http://onto.ui.sav.sk/agents.owl#Resource	NONE
NONE	http://onto.ui.sav.sk/agents.owl#Book	http://onto.ui.sav.sk/agents.owl#Goods	NONE
NONE	http://onto.ui.sav.sk/agents.owl#CD	http://onto.ui.sav.sk/agents.owl#Goods	NONE

Fig. 6: Example of storing ClassificationNodes and ClassificationSchemes into the table Name

```
<Association id="DataType-title"
    associationType="contains"
    sourceobject="Book"
    targetObject="http://www.w3.org/2001/XMLSchema#string"
/>
```

Fig. 7: ebXML RIM DataTypeProperty mapped construct example

lang	value	parent	charset
fr	http://onto.ui.sav.sk/agents.owl#title		NONE

Fig. 8: Example of storing Associations into the table Name

id	accessPolicy	objectType	associationType	sourceObject	targetObject	isConfirmedBySourceOwner	isConfirmedBySourceTarget
http://onto.ui.sav.sk/agents.owl#title	NONE	DataType	DataTypeProperty	-736d444a:1483b5f76e8:-7ff6	string	YES	YES

Fig. 9: Example of storing DataTypeProperty Association's type

The storage of the subClassOf Association's type is proceeded by specifying the objectType as a subClassOf and the sourceType and targetType as source and target of the association.

Table ClassificationNode will store all the ClassificationNodes and ClassificationSchemes objects and the heritage links between them. The ClassificationSchemes will be presented like simple ClassificationNodes and ClassificationNodes like ClassificationNodes with the entry Parent set at the parent of the ClassificationNode.

B. Relational Database Pre-stored procedures for better Business Repository semantic interpretation by the QM

For second step, we propose a set of pre-stored SQL procedures into the ebXML Repository Relational Database that will be used by the QueryManager to exploit the semantic of the data stored into the repository.

We present in –Fig. 10, Fig. 11, Fig. 12 - a sample of three SQL pre-stored procedures findDataTypeProperties, findSuperProperties and findTransitiveRelationships, this pre-stored procedures will be involved in interactions between the QueryManager and the repository database.

We note that additional pre-stored procedures can be added manually by Web Service administrators to express a specific semantic of the Business Domain.

```
CREATE PROCEDURE findDataTypeProperties (IN
className VARCHAR(100))
BEGIN
SELECT C.*
      FROM association A, name N, classificationnode C
      WHERE A.associationType LIKE 'subClassOf' AND
      N.value LIKE className AND
      A.targetObject = N.value AND
      C.id = A.sourceObject
END
```

Fig. 10: findDataTypeProperties of a specific ClassificationNode pre-stored procedures

```
CREATE PROCEDURE findSuperProperties (IN
propertyName VARCHAR(100))
BEGIN
SELECT A2.*
            FROM association A1, association A2, name N
            WHERE A2.associationType LIKE
'subPropertyOf' AND
            N.value LIKE propertyName AND
            N.value = A1.id AND
            A2.sourceObject = A1.id
END
```

Fig. 11: findSuperProperties of specific Association pre-stored procedures

```
CREATE PROCEDURE findTransitiveRelationships (IN
propertyName VARCHAR(100))
BEGIN
SELECT A2.*
            FROM association A1, association A2, name N
            WHERE A2.associationType LIKE
'transitiveProperty' AND
            N.value LIKE propertyName AND
            N.value = A1.id AND
            A2.sourceObject = A1.id
END
```

Fig. 12: findTransitiveRelationships of specific Association pre-stored procedures

VI. PROPOSED TOOL

For this purpose, we developed a tool that allow us to store the mapped OWL constructs to ebXML RIM following the workflow –Fig. 16- presented in our last article into the ebXML repository, and to execute pre-stored procedures over the resulting Relational Database.

The workflow of our tool that is an extension of the tool presented in our last article[1], starts with Names conversion, after that Associations conversion and finaly ClassificationNodes and ClassificationSchemes conversion. Workflow and screenshots of the tool are presented in figures –Fig. 13, Fig. 14, Fig. 15, Fig. 16 and Fig. 17-.

This tool is a Java application implementing the framework Jena for manipulating ontologies, and jdom API for parsing and writing XML file. The database we used here is a MySQL datababase.

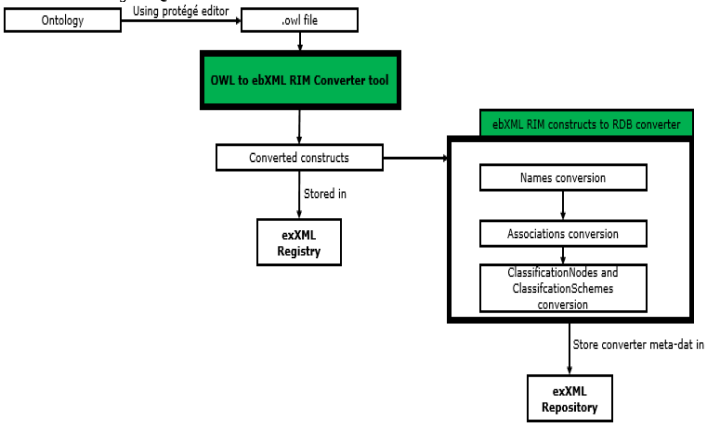

Fig. 13: Global workflow for enhancing the discovery of Web Services

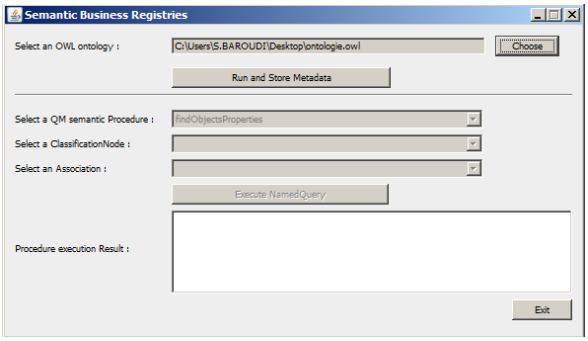

Fig. 14: Storing the OWL semantic data into the Business Repository and creating the pre-stored procedures

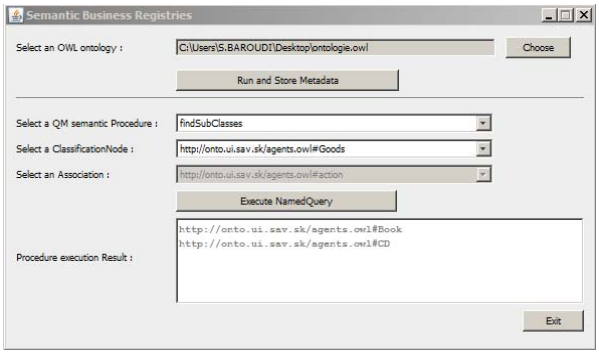

Fig. 15: Example of execution of the findSubClasses pre-stored procedure

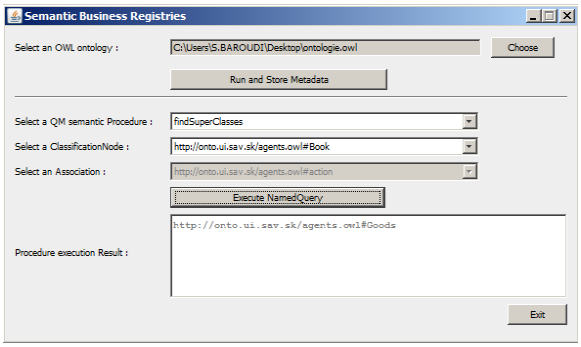

Fig. 16: Example of execution of the findSuperClasses pre-stored procedure

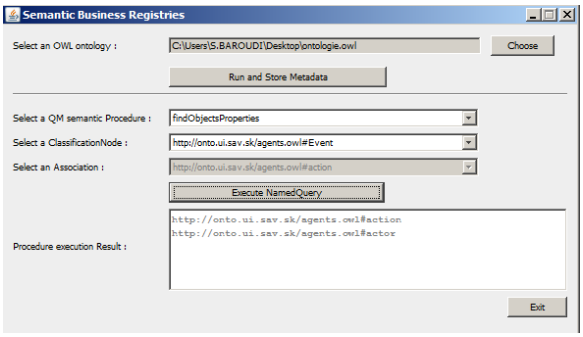

Fig. 17: Example of execution of the findObjectsProperties pre-stored procedure

VII. BACKGROUND AND RELATED WORK

An important trend in the evolution of the World Wide Web (WWW) has been the standardization of the Web Service technologies, by which a service provider can make a service available to consumers through an advertised protocol. The function of a Web Service is often to supply information, but can also involve exchange or sale of goods or obligations. Web Services provide a basis for interoperability between service providers and consumers, based on the reliable exchange of messages.

For the purpose of making the service discovery more easier and automated, many practices has been developed for introducing semantic into the business registries on registries based domains by storing meta-data in it registries. Mapping the domain ontology and using OWL-S [12] for constructing the meta-data are the most popular and developed practices.

In this article, we propose a new approach to rich a high stage of semantic by making the business registries a real OWL aware. This approach consist on storing the semantic ebXML RIM constructs into the Business Repository Database, and designing pre-stored procedures for the interpretation of these semantic by to QueryManager that is the main component for the communication between Web Services.

VIII. CONCLUSION AND OUTLOOK TO FUTURE WORK

Ontologies have been a research topic addressed in a rather small research community. This has changed drastically in the late nineties by the insight that a conceptual, yet executable model of an application domain provides a significant value. The impact has increased with the Semantic Web initiative and the Web Ontology Language (OWL).

The importance of semantics is also considered in the Web services area and there have been several efforts to improve the semantics support for Web services.

In this work, we tried to maximize the benefit of using the ontologies into the Business Registries by storing the mapped OWL DL semantic data to ebXML RIM specifications into the Business Repositories. To complete the semantic reasoning of these Business Registries, we designed and stored SQL procedures into the Business Repository Databases. These procedures will provide to the QueryManager the interpretation capacity of the semantic data stored in the Business Repository. To demonstrate the benefit of our approach, we developed a JAVA tool to do the storing process and to execute developed queries over this Database to evaluate our contribution.

As a future work, we intend to think about an advanced schema for the Business Repository Database to englobe semantic data and physical data of the Web Services.

REFERENCES

[1] Baroudi S., Bahaj M., Integrating Ontologies into ebXML Registries for Efficient Service Discovery, International Journal of Applied Engineering and Technology, pp 17-31, 2014.

[2] Romin I., An Introduction To ebXML, Web Services Business Strategies and Architectures, pp 220-235, 2002.

[3] Antonio Pereira A., Cunha F., Malheiro P., Azevedo A., EBXML - Overview, Initiatives and Applications, Innovation in Manufacturing Networks IFIP - The International Federation for Information Processing Volume 266, pp 127-136, 2008.

[4] Gunatilaka M., Wikramanayake G., Karunaratne D., Improving B2B Transactions by Exploiting Business Registries, 6th International Information Technology Conference, 2004.

[5] D. McGuinness and F. Harmelen, OWL web ontology language, Semantic Web: Concepts, Technologies and Applications, NASA Monographs in Systems and Software Engineering 2007, Springer-Verlag London Limited, pp 81-103, 2007.

[6] G. Antoniou and F. Harmalen, Web ontology language: OWL, Handbook on Ontologies, Springer-Verlag, pp 76-92, 2004.

[7] Mcguinness D.L., Fikes R., Hendler J., Stein, L.A. DAML+OIL: an ontology language for the Semantic Web, Intelligent Systems, IEEE Volume 17, Issue 5, pp 72-80, 2001.

[8] Miller E., An Introduction to the Resource Description Framework, Bulletin of the American Society for Information Science and Technology Volume 25, Issue 1, pages 15-19, 1998.

[9] Je Z. Pan, Resource Description Framework, International Handbooks on Information Systems 2009, pp 71-90, 2009.

[10] Gagnon M., Description Logics, Reasoning Web. Semantic Technologies for Information Systems Lecture Notes in Computer Science Volume 5689, pp 1-39, 2009.

[11] Barry D., Database Concepts and Standards.
[12] Martin D. , Paolucci M. , McIlraith S. , Burstein M. , McDermott D. , McGuinness D. , Parsia B., Payne T., Sabou M., Solanki M., Srinivasan N., Sycara K., Bringing Semantics to Web Services: The OWL-S Approach, First International Workshop, SWSWPC, Springer-Verlag Berlin Heidelberg, pp 26-42, 2004.
[13] Dogac A., Kabak Y., Laleci G., Enhancing ebXML registries to make them OWL Aware, Distributed and parallel DataBases, Springer Science + Business Media, pp 9-36, 2005.
[14] Cassio V. S. Prazeres, Maria D. G. PIMENTEL, Toward semantic web services as MVC applications from OWL-S via UML, Journal of Web Engineering, Vol. 9, No. 3 pp 243–265, 2010.
[15] Dogac, A., Kabak, Y., Laleci, G.B., Enriching ebXML registries with OWL ontologies for efficient service discovery, Web Services for e-Commerce and e-Government Applications, Proceedings. 14th International Workshop, pp 69 - 76, 2004.

AUTHORS

S. Baroudi was born in 1988, in Morocco. He is a Network and Security engineer in a French company in Casablanca and a PhD student in the Department of Mathematics and computer science, FSTS, University Hassan I, Settat, Morocco. His area of interest includes web ontologies and semantic web.

M. Bahaj was born in 1964, in Morocco. He got his PhD in Applied Mathematics, from University of Pau, France, in 1993. He is now working as a Professor at the Department of Mathematics & Computer Sciences, University of Hassan 1er, Faculty of Sciences & Technology Settat, Morocco. His research interests include pattern recognition, Semantic web & Ontology in MAS, Controls of mobiles agents

.