International Journal of Engineering Inventions
e-ISSN: 2278-7461, **p**-ISBN: 2319-6491
Volume 2, Issue 2 (January 2013) PP: 79-82

Ultra large scale systems to design interoperability

Mehdi Yaghoubi
Department of Computer Sciences, Faculty of Sciences, Golestan University, Gorgan, Golestan, Iran

Abstract:- Today by developing of the societies and increasing the enterprise and globalizing the industrial products, the communication among the organizations has remarkable growth. Each one of these organizations has its own special software. These softwares have several platforms and are implemented with different programming languages. Therefore the integrity and interoperability among Ultra large scale systems in secure environment are highly needed. The presence of heterogeneity in software industry has made the integrity and interoperability of the applications very hard and complicated. Therefore one cannot perform integration with traditional interactions and standards. Therefore a novel method of integration should be established. In this paper, we have examined the methods of establishing interoperability and then a suggested design would be offered. Our suggested design that is a three-layered model and has been built based on the web service technologies and semantic web, will remove the existing challenges (or it is a solution to remove the existing challenges).

Key words:- ultra large scale systems, interoperability, integration, agent

I. INTRODUCTION

Today most of large organizations have conflicts ab

out the need to integrate and interoperability in ultra large scale. An operable program should be able to have operations with other programs in hetero-genesis calculation environment. Despite all performed activities, still interoperability is a subject for which there are much debates. Before anything, some definition of interoperability would be developed:

- The ability of one system in using the other system resources [1].
- Efficient sharing, exchanging and integrating distributed data in heterogeneous information resource.
- Interoperability of the ability to combine business via software products to provide regardless the suppliers and products to provide reaching to the corporations and performance information using each identifiable user. Two main problems in establishing interoperability are autonomy and heterogeneity [2]. Indeed operable programs should have the ability to obtain information existing in heterogeneous and autonomous data resources.

A- Autonomy

The first problem in building interoperability is autonomy. Autonomy includes the system ability in performing the operations individually. Autonomy could be examined in several aspects: [1]. The types of autonomy are as follow:

- Communication autonomy: the system ability in making decision on whether to communicate whit other systems or not and that when to communicate.
- Execution autonomy: The system ability in performing local operation without interfering with other systems.
- Association autonomy: the system ability in making decision on whether to share its resources and functional with other systems or not.

B- Heterogeneity

The second problem of establishing interoperability is the presence of heterogeneity. Heterogeneity can be proposed in several aspects:

- Platform heterogeneity: i.e. heterogeneity in operating system or hardware.
- Data heterogeneity: comes from applied method to indicate the read-world facts.
- Semantic heterogeneity: expresses when there are differences in semantic, the way of presentation and use of similar or related data in the systems.

II. SEVERAL LEVELS OF THE SYSTEM INTEROPERABILITY

Several levels of the system interoperability include:

- Interoperability is signature level: examines the way of defining interface. That is comprised of operations and the order of interface parameters of a system. In order to perform this task interface definition language (IDL) is being used.
- Interoperability in protocol level: examines the way in which system methods are being requested. Web service choreography description language (WS-CDL) could be considered as an attempt in this regard.
- Interoperability in semantic level: deals with the problem of common understanding among service producers and its applicants. Using ontology and semantic service description framework like OWL-S, WSMO and METEOR-S [3], we can consider this subject. Interoperability in semantic level is important when we deal with the service interoperability in wide calculation environment.
- Interoperability in quality level: on one hand quality of service interoperability presents the structure form of quality needs of a service applicant and on the other hand it indicates the quality attributions presented by the service producers.
- Interoperability in context level: points to the semantic structure presented by the service producers and semantic structure requested by the service applicant.

  Although the service Interoperability levels are much being used, but there are other important factors. Some of which are mentioned here:
- Interoperability in business domain: relates to Interoperability or the inner performance of two business system with each other that causes them to share their public concepts and processes with each other. Applications are examples of business domain in particular. Therefore the application Interoperability indicates the Interoperability in the business domain in particular.
- Platform Interoperability: indicates Interoperability of the main middleware via two applications that have properties like using types of homogeneous data (such as numbers having same format and accuracy), interface details mechanism (like interface definition language) or the architecture styles (like using message oriented communication styles).

III. USING ONTOLOGY TO INTEROPERABILITY

By standardizing the format and the structure of the interfaces and of the exchanged messages, Web services provide interoperability among distributed and heterogeneous applications and platforms. However, apart from this interoperability at the syntactic level, semantic interoperability is also a crucial, and often even more challenging, requirement. Semantic heterogeneity refers to the intended meaning of the information. In order to achieve semantic interoperability in a heterogeneous information system, the meaning of the information that is exchanged has to be understood across the communicating subsystems. Three main causes for semantic heterogeneity are usually identified: (a) "confounding conflicts", which occur when information items seem to have the same meaning, but differ in reality; e.g. owing to different temporal contexts; (b) "scaling conflicts", which occur when different reference systems are used to measure a value; e.g. different currencies or different date formats; and (c) "naming conflicts", which occur when naming schemes of information differ significantly (a frequent phenomenon is the presence of homonyms and synonyms.)

The schema of a data source describes the way that data are structured when stored, but does not provide any information for their intended semantics. Therefore, metadata are required to allow for the understanding, management, and processing of these data. Using domain ontologies, it is possible to semantically annotate the involved data sources and infer mappings between them. Exploiting the information conveyed by the ontology and the annotations, it is possible to provide a measure indicating how close semantically the information provided by two data sources is. Furthermore, the use of ontologies allows to identify and construct, in a semi-automatic manner, the processes required to clean and reconcile data coming from different sources.

In the recent years, research efforts towards the realization of the Semantic Web vision have lead to the standardization of ontology languages such as RDF(S) and OWL. These standardization efforts have further facilitated the development of tools for creating, maintaining, and reasoning with ontologies, such as the Protege ontology editor or the Pellet OWL reasoner. Regarding the use of ontologies in information integration, several works have been proposed in the literature, which can be classified in three broad categories: single ontology approaches, multiple ontologies approaches, and hybrid approaches. A single, "global", ontology simplifies the integration process, but it is difficult to create and maintain, especially in the presence of changes in the data source schemas. On the other hand, multiple ontologies provide flexibility; however, comparing the sources becomes considerably more difficult. In hybrid approaches each source is described by its own ontology, using terms from a global, shared vocabulary.

IV. INTEROPERABILITY VIA SEMANTIC WEB SERVICE

Combining the advantages of both worlds, Web services and ontologies, has lead to the development of Semantic Web services. These are Web services that are semantically described, i.e., the parameters in their descriptions are annotated by concepts from associated domain ontology.

In particular, three main approaches have been proposed for bringing semantics to Web services: OWL-S, WSDL-S, and WSMO. The main idea in all these approaches is to use appropriate ontologies to semantically annotate the various aspects of a Web service description, such as inputs,outputs, preconditions, and effects, as well as non-functional parameters (e.g., QoS parameters). With Semantic Web services interoperability and integration is further facilitated. OWL-S is ontology for describing Semantic Web Services, built on top of the Web Ontology Language (OWL). Consists of three sub ontologies, describing, respectively, service profiles, service models and service groundings, which correspond to different aspects and levels of granularity of service descriptions. In general, the service profile is useful for service advertisement and discovery, while the model and the grounding provide information about how an agent can use a discovered service. A more detailed description of these parts is given below [4].

A service profile gives a high-level description of what the service provides to its clients. Hence, it is used during matchmaking to determine whether the service meets the client's needs. The functionality of the service is specified in terms of inputs and outputs, as well as preconditions, which are requirements that the client should satisfy in order to use the service, and effects that may result from the service execution.It also contains information about other features of the service, such as the entity or the organization that offers it, the category of the service in a given classification system or quality ratings (response time, reliability, etc.).

A service model describes how the service works, presenting it from the perspective of a process. Given a request, a process may either return some information, specified by its inputs and outputs, or produce a change in the world state, specified by its preconditions and effects. In addition, a process may be either atomic or composite. In the first case, there is a single interaction between the client and the service, i.e. the client sends a single request and receives an single response. In the second case, there is a series of interactions, i.e., of exchanged messages, with the service maintaining some state throughout it. OWL-S provides a set of control constructs for specifying composite processes, such as sequence, split, if-then-else, repeat-while. The specification of data flow between the sub-processes is supported, as well.

Finally, service grounding provides information about how to access and interact with the service, such as the communication protocol to be used and the structure of the exchanged messages. Essentially, it grounds the service description to a concrete implementation. This is achieved in conjunction with WSDL, which has been chosen due to its widespread use in industry for describing Web services. For example, an OWL-S atomic process is mapped to a WSDL operation.

V. SUGGESTED DESIGN

One of the problems in software engineering in ULS system designs is lack of proper approach in these systems width, such that the problems like integration, documentation and coordination among the minor parts of the systems would not be satisfied. Hare a model is presented that has three layers. Two layers of which are defined a large scale system components and third layer would be spread on all the components widely and give it single view, so these three layers solve the intended challenge. The first layer of this model that could be called follow-up layer is established based on web service technology and the second layer that could be called definition and documentation layer provides the possibility of comparison and coordination semi-autonomously to apply follow-up layer possibility. This layer is comprised of the files in XML frameworks defined on follow-up layer and present a semantic definition of follow-up layer possibilities with its details. Third layer searches some information of the second layer using a semantic web search motor and provides the possibility of connection among. These components and also forms possibilities to focused management with distributed database in this layer. In this layer, facilities like establishing and controlling contract and completing the component security parts would be formed.

A) First layer

In this layer facilities defied in the second layer would be defined by a set of classes, methods and attachments and allow definitions and adjustments in the second layer to be automatic by this layer and provide connection of other ULS components in the lowest level. This level is a set of methods, classes and attachment that are in close relation with related component database and provides just a set of facilities to edit and reach to the information of this component.

B) Second layer

In the second layer that could be considers a knowledge base of defined concept and facilities in the first layer, there is a layer based on XML that defines the layer facilities details and the way of using semantic and rules and regulation of using these facilities. Also in this layer there is a set of control classes to conclude contracts and use the component facilities. In order to clarify the presented issues in the second layer the following code piece shows some of the presented information in this layer.

```
< Component Name=”Educational system” Version=”1.1.0” Production Date=”2005/01/01”
lastUpdate=”2010/01/01” DatabaseType=”SqlServer2005”>
<Utilitys>
<Utility Name=”studentList” Type= report CreateDate=”2006” programmerId=”10”  />
<Utility Name=”studentRegistration” Type= form CreateDate=”2006” programmerId=”11” />
<Utility Name=”studentSelectionCourse” Type= Query CreateDate=”2006” programmerId=”8” />
</Utilitys>
<programmers >
<programmer Id=”10” Name=”Alex” phone=”09354012234” Position=”administrator” />
<programmer Id=”11” Name=”Martin” phone=”0936789543” Position=”User” />
<programmer Id=”8” Name=”Kevin” phone=”0935401223” Position=”User” />
</programmers >
</Component>
```

In the above code piece a component named instruction system has been introduced in which information like the system name, the copy number, date of manufacturing product, last update and type of database related to it have been written in XML framework. Also in this section helpful components used in this program have been indicated. For example a component called student register that is a type of that form and has been written by programming with indicator 11 and also in tag section related to the programmer has been introduced. The intended agent facilities interoperability and communication by referring to this information existed in metadata and extracting needed information and reporting this information to the intended receiving service.

C) Third layer

In third layer there is a semantic search engine (semantic web is evolved generation of syntax web (traditional one) that enables machines (computerized agents) to receive web content as human recognizes) that uses defined information of first and second layers and suggests initial comments to connect several components and obtain information and conclude it to the user. It should be mentioned that the way of searching in semantic web is such that in syntax web (traditional one) search is based on keywords and just information is retrieved that the intended word would be in it or repeated for many times, even if its semantic be different, however in semantic web, search would be done based on the semantic and it exactly seeks the same thing intended by the user.

VI. CONCLUSION

It has been said that there are many problems on the way of establishing interoperability among the enterprises applied programs in ultra large scale that come from the nature of these kinds of systems. Many authors seek to solutions for this difficulty. In this paper we consider to examine some of these approaches. Using services was one of the methods that have reached to this in ultra large scale in several levels of the service. The other method is to use ontology to improve the integration in ULS system levels that have solved this problem by using simple, multiple and combined ontology approaches. In the following by combining advantages and benefits of web services and ontologies a method called semantic web service has been formed to establish interoperability in ultra large scale. Using the service general model as the basis of developing middleware and proper languages that load to forming the service interoperability was another method that has been introduced in this article. Also the role of ontology and semantic web in the improvement of interoperability has been discussed. At last we offered a three-layered method. Two layer of which are defined on ULS system components and third layer is distributed widely on all the components and give it a signal view. Regarding that this alternative uses a multilayered architecture, it gives high portability to the users. Meanwhile this approach uses the other methods advantages. This alternative leads to the improvement interoperability in ultra large scale. However, in spite of all the activities done in technology field, interoperability is still a subject for which there are many debates.

REFERENCES

1) Frankel.D.S. (2003). Model Driven Architecture: Applying MDA to Enterprise Computing. OMG Press, Wiley Publishing.
2) John A. Bellcore.M. (1992), Large Scale Interoperability and Distributed Transaction Processing,0-81862697-6/9203.00 IEEE.
3) Athanasopoulos.G, Tsalgatidou.A, Pantazoglou.M. (2006). Interoperability among Heterogeneous Services IEEE International Conference on Services Computing.
4) Sellis .T, Skoutas .D, Staikos .K. (2008).Database Interoperability through Web Services and Ontologies, 8th IEEE International Conference.