2010 10th Annual International Symposium on Applications and the Internet

Service Level Semantic Interoperability

Asad Masood Khattak, Zeeshan Pervez, A. M. Jehad Sarkar, Young-Koo Lee
Department of Computer Engineering, Kyung Hee University, Korea
{asad.masood, zeeshan, jehad}@oslab.khu.ac.kr, yklee@khu.ac.kr

Abstract*— Interoperability is a collaborative and multifaceted task to overcome the problems of incompatibilities among organizations, structures, data, architecture, services, and business rules. To achieve interoperability, the aspect for interoperability and requirements for applications and services must be known. SC^3 provides Ubiquitous Life Care (u-Life care) for robust healthcare services, service suggestions, and change in system behaviour for better care with low cost. Focus of this paper is on service level semantic interoperability for u-life care to overcome the limitations of publishing the service schema, expose all the service level interfaces, and hardcode data interpretation and manipulation by extending the standards of HL7 protocol for domain specific services. Service level semantic and process level interoperability framework architecture is proposed to achieve proper communication of clients and third party services with SC^3. By incorporating semantic interoperability with the use of ontology, we reduced the interoperability stack to three levels. This helped in reducing its complexity and helped in intelligent processing of request and redirection of request to appropriate hosts.

I. INTRODUCTION

Interoperability is the ability of any two or more entities to communicate and exchange information meaningfully even with the differences among them [1]. Achieving interoperability among applications and services is a challenging task that involves lot of technological aspects to be resolved [2]. Technically, message should properly be transported to the destination. Semantically, message contents should be understandable for the receiver as it is for sender [3]. The technical interoperability is mostly achieved using CORBA, SOA, and Web Services, while semantic interoperability still remains the gray area and needs to be unfolded. For message contents well known techniques of tightly coupled model for interaction were used [1]. These worked well, were cost-effective, and easy to deploy, but the main issue was service evolution due to new changes.

To maintain quality and availability level of life care services with powerful, flexible, and cost-effective infrastructure for life care services that can fulfil the vision of *ubiquitous life care (u-life care)* is required [4]. For this, we have developed a platform architecture, called Secured Wireless Sensor Network (WSN) - integrated Cloud Computing for u-Life Care (SC^3) [5] that supports a number of proprietary services with two aims. Firstly, the services are internalized but perhaps having significant storage and computing time requirements in which case these services are available for distributed application on multiple SC^3-mounted platforms. Secondly, the services are externalized so that subscribed users may connect with and use these for their own purposes. Examples of supportable services include real-time home care and safety monitoring services, information sharing, emergency connection services, and patient monitoring and care services [4]. The security and privacy of user health data is maintained by SC^3 having internal security firewall.

Services in distributed instances of SC^3 as well as third party services communicate together. For their proper communication even with their autonomous and heterogeneous nature, the proposed Service Level Semantic Interoperability (SLSI) makes them communicate properly. An ontology for message composition and interpretation that facilitates in achieving semantic interoperability is developed. It is used to understand the inner contents of message. The service interoperability stack has been modified from five levels to three levels by incorporating ontology to handle most of the heterogeneity issues. This provides low cost, simple, more flexible, service on demand, less communication overhead, and smarter processing of user request. SLSI demonstrate the process of less communication overhead, no schema publishing, and no interface exposition to the outer applications. Using SLSI, SC^3 share and access services from other service providers in more real-time manner.

This paper is arranged as follows: In Section II we demonstrate our modifications in service interoperability stack. Section III is about the proposed semantic service level interoperability framework architecture. In Section IV the advantage of SLSI for less communication overhead is discussed. In Section V we conclude our findings.

II. INTEROPERABILITY STACK

To achieve interoperability among services, application developer need to know open requirements that cause problems during service interaction. A set of five level aspects with their requirements specifications are identified in [3] that needs to work out for the purpose of achieving interoperability. The author in [3] developed an ontology for end to end communication as well as the interaction among services.

The requirements levels to achieve interoperability are formalized by incorporating ontology. The division in levels is based on entities involved in these levels. The base level is the *Technology Level* (see Fig 1) where technical interoperability is achieved for the purpose of agreeing on or accepting particular encoding scheme for requests and response, selection of communication protocol, selection of possible

978-0-7695-4107-5/10 $26.00 © 2010 IEEE
DOI 10.1109/SAINT.2010.54

Fig. 1. Optimized Interoperability Stack

middleware, language, and the platform for working environment. The encoding and connectivity requires the specification of language, adaptors, middleware, and organization service environment. At *Business Level* different business strategies are imposed on the services, regulations on the services, and mode of use. Business organizations have different business strategies depending on their location and scope of business and to work on these they define and regularly refine their rules. This level is quite variable in nature but is less effective on overall interoperability process.

The *Semantic Level* is the level that we merged (a) Service, (b) Community, and (c) Organization Levels of stack designed in [3]. Their focus is more on functions calls and response for consumers request rather than dealing with the inner details of request by incorporating semantics in the message contents. Our extensions in ontology incorporate the explicit semantics in message contents. These semantics are used in recognition of requester intentions by interpreting the message contents. The ontology will be used by all the service providers and will compose their request and response messages in accordance to that ontology. For this, we have extended the standard vocabulary of Health Level Seven (HL7) [6] that is used as a protocol for message composition and transportation and the ontology presented in [3]. From message contents using ontology, the service can interpret the requesting application or service requirements and in response show the appropriate behavior. Here, the incompatibility (syntactic and semantic) is detected first and then appropriate conversion procedure is followed to convert these messages to a compatible form. For response the message is converted back to its original form. Conversion is possible on both ends of request or response as this stack is used on both sides. It also helps in loose coupling among services and their functionality for implementation.

This service level semantic interoperability stack eliminates some of the communication overhead. It only needs to send the request and wait for response rather than sending request for available interface(s) and the schema(s) separately for the service to both UDDI (i.e., Universal Description, Discovery and Integration) as well as the service providers. It is more cost effective and flexible for dynamic environments where requester and responder are not concerned with the schema on the other sides.

III. Service Interoperability Framework

SC^3 services, due to its cloud-based environment, may be accessed via standard web service approach i.e., sending and receiving SOAP (i.e., Simple Object Access Protocol) messages while using service information from UDDI. However, because of the two ways in which the SC^3 services can be deployed, service discovery and service availability communications can be optimized for service consumption. For this we propose Service Level Semantic Interoperability (SLSI) Framework that works to achieve semantic interoperability and eliminate the tight coupling among services. In this case applications only need the information about the services provided by a particular web service. Then a message is composed using a standard messaging protocol, communicated among web services and application for consuming the services. The tight integration of all the components makes the semantic interoperability possible (see Fig 2). In general, this architecture is applicable in all available web services, but in our case we have used semantic interoperability in our SC^3 system to provide and avail different services on subscription bases. Working detail of these components is as given:

A. Request and Response

Responsibility of this module is to actively listen all incoming as well as outgoing requests and responses from/to outer world web services and applications. Its second responsibility is to properly navigate the request and response to appropriate modules within the service. For this matter Request and Response module coordinate with its adjacent modules.

B. Interaction

A complete track of interaction taking place for service requests made by another service or application is maintained in this module. The service or application request may be partially or fully completed by a service on the local platform; if partially, the remaining parts of the service request are intelligently sent to other services or SC^3 platforms that can fulfil the needs of requester. So in this case all the indirections followed for the completion will be maintained in this interaction. This interaction is also used to answer back the consumer in an efficient manner and also maintain a complete history (of only subscribed consumers) of every interaction with details for future reference in case required. It initiates a thread for each request and response. Maintain information about all the indirections for part of a message as well as for a complete message. Different roles are assigned during interaction that are useful for maintain the history properly.

InitiatorRole: is responsible for maintaining the history of interaction initiator. *SenderRole:* Each interaction has a sender role as there will be a sender of request for the interaction or this service might be the requester. *FulfillerRole*: is designated to fulfil the request of requesting service or application. *ReceiverRole:* for request of this service fulfilled by another service, the *ReceiverRole* receives the responder information.

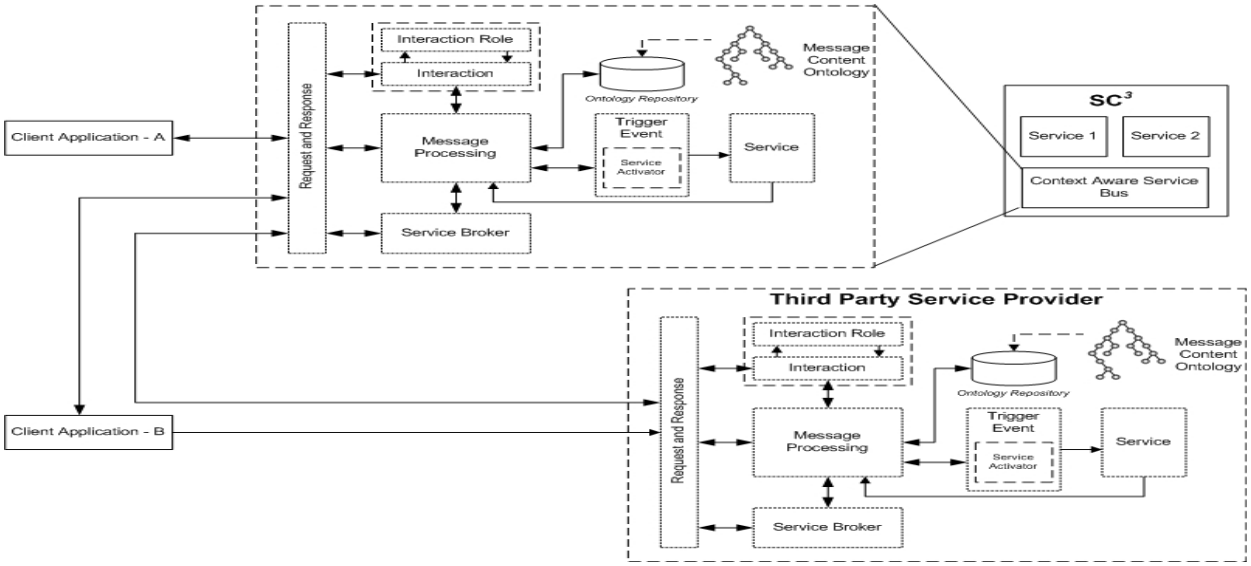

Fig. 2, Service Level Semantic Interoperability (SLSI) Framework Architecture

C. Message Content Ontology (MCO)

It is a semantic structure, containing all the information regarding Interactions, Message Types, organization rules, and Trigger Events (see Fig 3). It is designed to provide a standard structure for composing a generic message that can be communicated and interpreted on any service. The HL7 [6] document is used as a baseline for the development of Message Content Ontology (MCO). It is required to address any query about interaction or checking the message status that the message is a request or response of a particular request. MCO contains all the process artifacts, broker information, message composition and decomposition, interaction information, business rules, and trigger events see Fig 2. It describes how they are associated with each other and helps in automation as well as smart interpretation of message contents. After finding out from the message contents suitable service request, an appropriate service is activated.

MCO is also used to restructure partial messages to be sent to other service providers while maintaining the semantics of the existing message. When the messages are received on the other side then MCO is used again to interpret the contents of message and direct the message for appropriate service on that particular platform.

D. Message Processing

Message Processing is one of the most important components to achieve semantic interoperability. When message is transferred from Request and Response module to Message Processing module then it is intelligently parsed for its intended service. The message type for request and response is recognized in this module using MCO. In this component we use *Pellet* (an inference engine) to infer the intention of message. If the requested service is available then the Trigger Event module is activated that initiates a thread of that service, else it is directed to Service Broker that identifies appropriate service for the request.

E. Trigger Event

Once the intent of message is identified then the required service is activated with the help of Service Activator in Trigger Event. It maintains complete session information for the service execution till the job is completed and then the results are returned to message processing module. If there are some more results for the consumer then all these are combined together and a response message is composed which is then returned.

F. Service Broker

This module stores information of all domain relevant services made available by particular service provider. In case if some particular service demanded by consumer is not available then the service broker is used to direct the consumer request to its intended service. Information about consumer's intention is detected in Message Processing module. So, consumer is not required to contact UDDI again for the required service search. This eliminates the communication overload for service discovery using UDDI.

IV. Message Redirection

One of the most important features provided by Service Level Semantic Interoperability (SLSI) is that the communication load is reduced for service discovery at client side using semantic interpretation of the message contents. When any service of SC^3 or any other service using SLSI receives a request from an application or another service for a job, then the job is completed on that service provider's server, but in case the required service is not available then this current service will redirect the request to another service provider rather than the requester searching again for service.

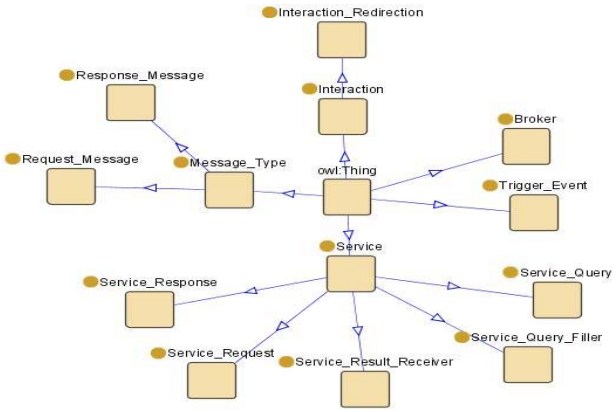

Fig. 3, Class view of Message Content Ontology.

The actual service search and consumption as well as the modified one with the help of SLSI are shown in Fig 4. Suppose, an application need to consume two services and both services are not provided by a single provider then the consumer will request twice for service discovery from UDDI as shown in Fig 4 (black lines). While in SLSI, it only need to search for one service and all related (domain specific) services are indexed in service broker. So in message processing module, SLSI will detect that the consumer also need to consume another service that is not available here so it will direct its second service request to the service broker and that will direct the request to appropriate service provider (see Fig 4, blue lines).

The results of requests are returned in two ways, (1) if the request is a part for previous request then it is returned to redirection node and the end results are combined together and returned to consumer. (2) For a simple and single request redirection, the results are simply returned to the consumer rather than the redirection node. To achieve the facility of redirection, a complete list of available related (domain specific) services needs to be maintained. To detect the message requirement proper interpretation of message contents is important. This model also reduces the cost for availing the services.

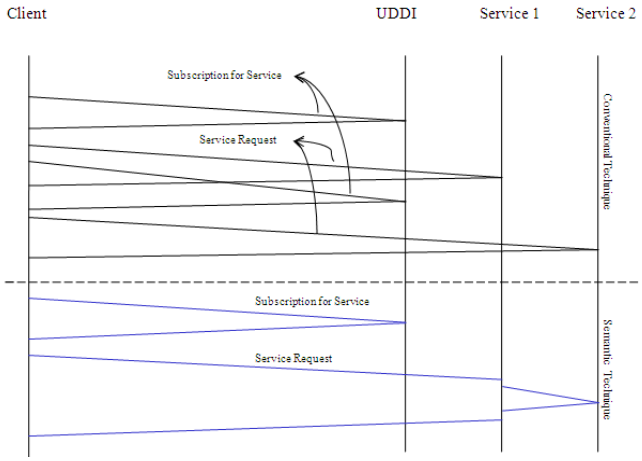

Fig. 4. Service discovery and consumption comparison for communication overhead between traditional technique and SLSI.

V. Conclusions

Service Level Semantic Interoperability (SLSI) framework has been proposed in this paper. To provide the semantic interoperability, we have worked on service interoperability stack. The proposed SLSI architecture is smart enough to process message and understand the contents of message and then respond accordingly. With the help of SLSI, we have reduced the communication overhead among UDDI, service consumers, and service providers. Currently we are working on refinement of message content ontology as well as the implementation of proposed system. The future intentions are to work more precisely on service broker for efficient and accurate communication for request and response. In future we are also focusing on issues like service communication from diverse domains. Considering different parameters like, service updates and service migration are also in the pipeline.

VI. Acknowledgement

This research was supported by the MKE (Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program supervised by the NIPA (National IT Industry Promotion Agency)" (NIPA-2009-(C1090-0902-0002)). This work was supported by the IT R&D program of MKE/KEIT. [2009-S-033-01, Development of SaaS Platform for S/W Service of Small and Medium sized Enterprises].

References

[1] D. Konstantas. Object oriented interoperability. In ECOOP '93 - Object-Oriented Programming: 7th European Conference, volume 707 of LNCS, pages 80–102. Springer-Verlag GmbH, 1993.

[2] A. Bracciali, A. Brogi, and C. Canal. Dynamically Adapting the Behaviour of Software Components. In F. Arhab and C. Talcott, editors, COORDINATION 2002, volume 2315 of LNCS, pages 88–95. Springer-Verlag Heidelberg, 2002.

[3] T. Ruoklainen, and L. Kutvonen, "Interoperability in Service-Based Communities". In Business Process Management Workshops: BPM 2005, C. Bussler and A. Haller, Eds., vol. 3812 of Lecture Notes in Computer Science, Springer-Verlag, pp. 317–328, Nancy, France, 2005.

[4] A. M. Khattak, P. T. H. Truc, L. X. Hung, L. T. Vinh, V. H. Dang, D. Guan, Z. Pervez, M. Han, S. Y. Lee, and Y. K. Lee, "Context-aware Human Activity Recognition and Decision Making", 12th IEEE International Conference on e-Health, Networking, and Application and Services, France, July 2010 (Accepted for Publication)

[5] L. X. Hung, P. T. H. Truc, L. T. Vinh, A. M. Khattak, M. Han, D. V. Hung, M. M. Hassan, S. Y. Lee, Y. K. Lee, E. N. Huh, "Secured WSN-integrated Cloud Computing for u-Life Care", 7th IEEE Consumer Communications and Networking Conference (CCNC), USA, 2010.

[6] Health Level Seven (HL7 V3), Meta-Model Version 1.14, http://www.hl7.org/library/data-model/met/C114/met0114l.pdf