Resolving Interoperability Issues of Date with Null Value and Collection of Complex Data Types by Using JADE-WSIG Framework

Jaspreet Chawla
Department of Computer Science & Engineering, JSS Academy of Technical Education, Noida. Affiliated to AKTU, Noida, UP, India.
E-mail: jaspreetchawla1@rediffmail.com

Anil Kr. Ahlawat
Department of Computer Science & Engineering, KIET Group of Institutions, Ghaziabad. Affiliated to AKTU, Lucknow, UP, India.
E-mail: anil.ahlawat@kiet.edu

Received December 20, 2020; Accepted February 28, 2021
ISSN: 1735-188X
DOI: 10.14704/WEB/V18I1/WEB18088

Abstract

In today's world, the internet and distributed computing make things so convenient that web services can be easily built and fetch from any platform. Web services are loosely coupled, interoperable, and heterogeneous hence they help to connect web applications of different languages. Interoperability is the major factor while transferring web services from one platform to another platform.WS-I basic profile 1.0/1.1 organization provides guidelines to achieve interoperability of web services at a basic level but still, issues arise at a complex level. In this paper, we have discussed and shown the result of two interoperability issues i.e. Date with Null value and Collection of Complex data types of web services using JAVA and.NET environment. In the first issue.NET treat NULL as a value type and JAVA treats NULL as reference type in date-time data type. So, whenever a JAVA client fetches a web service built in.NET it will show a parsable error. In the second issue, a web service data structure contains elements of any type. Whenever, a web service built with the 'ArrayOfAnyType' data structure, it can be easily mapped to a.NET client but not to JAVA client. Hence, a data type mismatch issue arises here. To resolve these inter-platform issues, we have used JADE-WSIG as middleware between web services and agent-technology.

Keywords

Web service, Date with Null Value, Complex Data Types, Agent Technology JADE-WSIG.

Introduction

Principles of web service enlighten the next generation of business applications. To build web-based applications, developers use different programming languages like JAVA, .NET, Python, etc. As these languages deploy on a different platform, so a sort of communication is required between them. Here web service comes in the picture. Web service provides a common platform for multiple web applications to communicate with each other. The aim of web service is to exchange data between client and server with the help of XML. Whenever the web applications pass data to each other, they are actually passing data that is stored in XML documents. The special characteristics of web service are loosely coupled, heterogeneous, distributed, and interoperable [Balmabrouk et al., 2016]. Therefore, in this paper, our main focus is on web service interoperable issues and it occurs when the different clients working on different platforms try to invoke a web service apart from its software and hardware configuration. An interoperable web service works across different operating systems, platforms, languages, and applications [Johari & Kaur, 2013]. W3C has provided three main standards to web service and these are SOAP (Simple Object Access Protocol), UDDI (Universal Discovery Description Integration) and WSDL (Web Service Description Language) [Balmabrouk et al., 2016].

SOAP: SOAP is an XML based messaging protocol that helps to pass information between different service client's applications.
UDDI: UDDI is a platform-independent, XML based open framework, like directory used to describe, search, and register web services.
WSDL: It is an XML based document help to describe the structure of web services. Basically it gives information to clients about what a web service does and how the information can be retrieved from a service.

As shown in Fig1, with the help of these components like UDDI, SOAP, and WSDL clients can fetch, search, register/deregister and update the web services from UDDI. In this paper, we are using web service that is built at.NET platform and is further used by JAVA clients, the issues of interoperability occur in this current scenario when a JAVA client tries to invoke the web service implemented in.NET environment. The important web services used worldwide are Google mail, weather forecasting, currency converter, etc.

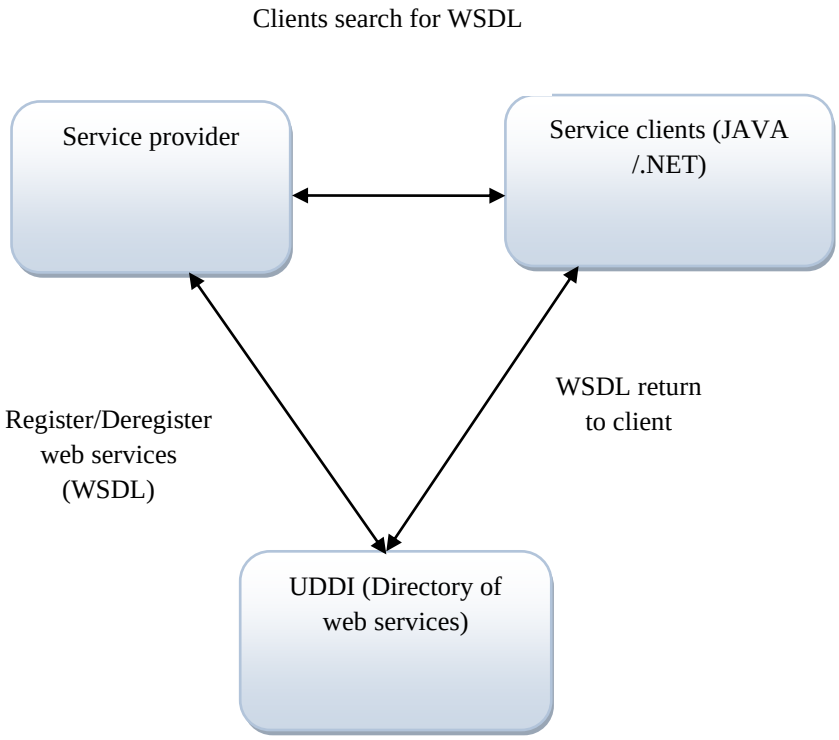

Fig. 1 A framework of web service

In spite of generic characteristics and standards of web services, it is still facing many interoperability issues [Haile & Altmann, 2018]. Web services build in JAVA or.NET platform is rapidly used in industries to develop enterprise-level application [Balmabrouk et al., 2016]. Interoperability features fail as the level of web service becomes more complex. WS-I (web service interoperability) [WS-I,2019] basic profile 1.0/1.1 organization has provided guidelines related to web services interoperability and proved that issues can be removed at an elementary level fully but only minimized at the complex level [Siddig,2015]. In today's world, if web clients want to host a web service publically, then all the clients of your business firm need to be used updated tool kits and modernized interface designs. Otherwise, a basic HTTP protocol, used for message passing of SOAP-based web services can increase the chance of interoperability [Ibrahim & Hassan, 2019].

There are a number of interoperability issues arises when exchanging data between JAVA and a.NET platform like a precision issue in data types, a null value in date data type, date-time precision issue, collection of complex data types, URI reference declaration as a namespace in WSDL, unsigned number issues, an array with a null value, etc [WS-I, 2007]. If any small change was done in the syntactic, semantic, or technical level of interoperability, the accurate results of service will not be invoked on the client-side [Sujala D Shetty *et al*., 2009].

Related Work

A web service interoperability issue has been the subject matter of many researchers. Due to boom in internet and more usage of mobile application, researchers are giving an important contribution to web services and how to make it easy access for an end user. According to Fabio Bellifeminea *et al*., 2008, the initial software development that initiated the JADE software was telecom Italia together with the University of PHARMA in July, 1998. They have explained well the framework for developing multi-agent applications with the help of JADE and how JADE work with proper functionalities and with less restriction imposing on it.

Beoum Sun Kim *et.al*, 2009 has focused on the smaller memory space usage and faster processing of quarry while searching in Directory Facilitator (DF) in JADE. Another step towards web services integration with WSIG was taken by Dominic Greenwood *et al*. they proposed a framework to integrating of web services with software agents using IEEE FIPA approaches and how FIPA standards incorporates the searching of web services and agents from their directories.

Sujala D Shetty *et al*., 2009 has explained well on web service interoperability and its related issues with experimental results but solution of issues are given in a brief manner. R. Udaya kumar *et al*., 2003 have given comparison of JAVA and.NET technology and explained which architecture are good to support web services and in building government application. But have not explained on interoperability issues and solutions of both the languages.

Ivano Alessandro Elia,2014 have done a huge setup of thousands of inbuilt web services, three major servers and 11 clients sub system and prove that interoperability always exist when transferring the web services from one platform to another. Service generation to service testing, at every step, a large number of error and warnings come across but how to resolve these errors were not discussed.

Jihad Mohammed Siddig, 2015 has done a comparatives study on the performance and evaluation of two technologies JADE and Aglet and proved that how jade has given better results to inter platform mobility. Recently a paper has been published by Federico Bergenti *et al.*, 2020 has explained and proved the importance of JADE from last two decades and still would be helpful in future projects. They have described well on building multi-agent, interoperable and complex system with the help of JADE. They have discussed that how JADE help in building critical system and real life- applications.

Petrosino, G *et al.*, 2019 focused on jade script and jade script ontology's. They also discussed and deployed the mapping of jade-script with structured and behavioral ontology's. Aarti singh et al., 2011 has discussed and done the comparison of different agent toolkits. Many researchers discussed about semantic and syntactic interoperability issues of web services in many applications.

Due to its flexible features and performance, we have adopted JADE with WSIG to minimize the interoperability issues of web services we are interested moreover in coordination and interaction among multi-agent system and web services. Client's results have been proved that how issues arise and resolved with JADE-WSIG. JADE is used to build agents as one or multiple agents are helpful in resolving the issues. Overall, Multi-agent system provide a heterogeneous and dynamic behavior among multiple systems and achieve desired goals [Nezhad et. al, 2006]. In our next paper; we will try to implement JADE on cloud web services.

Mapping of Web Service with Agent Technology

To remove these interoperability issues, we have integrated agent-technology with web service. Agent technology has been standardized by FIPA (Foundation for Intelligent Physical Agents) for developing specification of agent system [Huhns, 2002 & JADE, 2002]. The objective of FIPA is to standardize interoperability among software agents [Petrosino & Bergenti, 2019]. An agent is like an entity that is placed in an environment where it senses different parameters that help to achieve its goal goals like sharing data with other agents, involvement in the complex and distributed environment of distributed information system, etc. [Liu., 2017]. The agent platform helps in developing a middleware between multiple agents and web services [Chawla et al, 2018].

An agent system contains more than one agent that exhibits properties like flexibility, adaptability, autonomy, social, pro-active, mobility, benevolence, etc [Nwana & Ndumu, 1998]. Agents communicate with each other using ACL (Agent Communication Language) to share information and perform tasks [Ibrahim & Hassan, 2019]. ACL consists of language and vocabulary just like other high-level languages. Most agent system is written either in C/C++ or JAVA because a software agent mostly exhibits the characteristics of object–oriented programming [Kanhere & Raja, 2018]. The main purpose of agent technology is enhancing flexibility and improves the scalable system of web services. Today, numbers of tools are available in the market to support agent oriented software development and most of them support the SDLC life cycle that comprises of analysis, design, and development, coding, testing, and debugging activities

[Nwana & Ndumu, 1998]. In this paper, we have used a JADE tool that works as an agent to do the mapping between web services and multi-agent system. JADE used to build an interoperable multi-agent system (MAS) for the management of network information resources in compliance with FIPA specifications [Kamdar et al. 2018 & Nwana, 1998].

JADE is enormously flexible that it can be easily integrated with different programming languages like.NET and J2EE [Chawla et al., 2018]. Further, JADE also allows agents to Discover and communicate with others. Every agent in JADE is provided with AID (Agent Identification Number) to register /deregister its services. JADE supports the JAVA libraries, homogenous set of API and executable environment for the web services [Poggi, 2010 & Nwana, 1996]. It automatically supports data the format between agents and web services. As shown in Fig 2, there is an add-on of JADE called WSIG (Web Service Integrated Gateway) that acts as middleware with JADE, explores agent services, and coverts SOAP messages to ACL instructions. Like UDDI, DF (Directory Facilitator) is also a directory that contains information about agent's. All agents publish, update, and register their services in DF only [Haile & Altmann, 2018]. JADE work as a software agent and without changing specifications, it combines with WSIG to support bi-directional integration between web services and multi-Agent systems. The main objective of WSIG is to publish the same set of interoperable services in the UDDI directory as well as in Agent-DF [JADE,2002].

Fig. 2 JADE-WSIG as a middleware

There are ontology's that are used to access the agent services description defined in the DF and identify the actions that an agent is able to perform [Kim et al.,2009]. Fig 3 shows the mapping between Agent service and web service description. The ontology defines vocabulary and semantic of each language and helps to pass messages between them [Coria et al., 2014 & JADE, 2017]. Web service and agent both have their different operations and actions. JADE –WSIG defines its own ontology to do mapping of agents with web services. For each agent service description, WSIG automatically maps a web service (WSDL), whose operations are correspondingly matched to the actions of agents [JADE,2002].

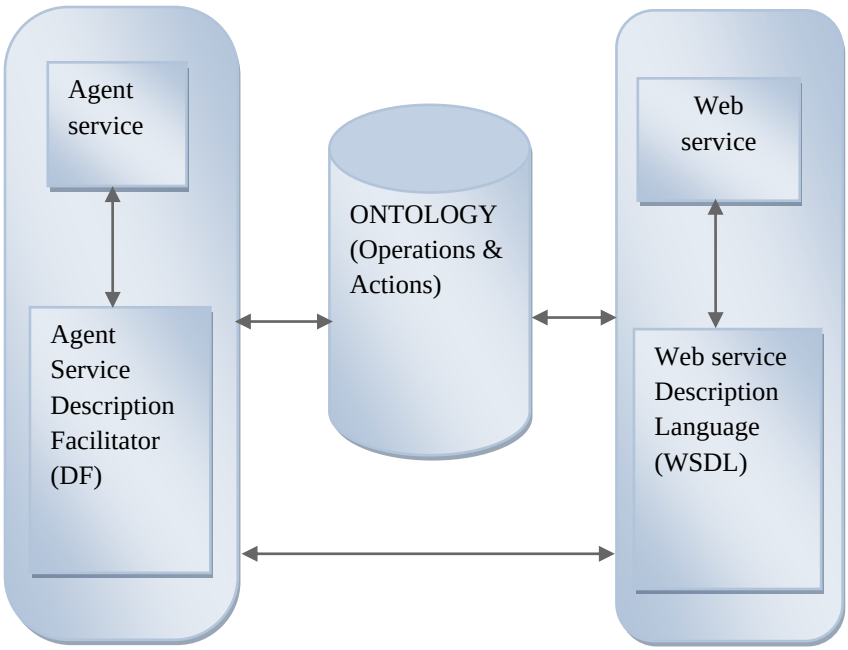

Fig. 3 Mapping between WSDL and DF [31]

In order to map an agent service with a web service, it's compulsory to keep WSIG property to "true" at the time of agent registration in DF [Kim et al.,2009 & JADE -2017]. All actions defined in the agent service description ontology's will automatically be maps to the web service description. Hence a message exchange is possible with the WSIG between two frameworks.

Issues related to Web Service Interoperability

Interoperability is the major concern while transferring web service between different platform clients [Elia et .al, 2014]. In this paper, we have created a SOAP-based web service named "service1.asmx.cs". when this web service is executed it will generate a WSDL file as shown in Fig 4. With the help of this web service we are showing the interoperability issues among different platforms.

SOAP message defines a standard way to send the data and methods to the remote client. The HTTP protocol is used for binding the SOAP messages [Ibrahim & Hassan, 2019]. Web service is using the well- established guidelines of W3C for implementing interoperability but sometimes they are not adequate to resolve the interoperability issues of service clients. Different platforms always have different data types, data formats, and standards. To show the interoperability issues of web services, we have created three clients 1) JAVA client 2) .NET client 3) JADE client. These three clients' works on two interoperability issue 1) Date with null value 2) collection of complex data types.

Fig. 4 Snapshot of web service 'service1.wsdl'

Date with Null Check Method

A web service always builds in XML language and it can be implemented in different platforms like JAVA, .NET, python, etc. We have chosen the.NET(C#) environment to implement the web service "service1.asmx" In XML, there is a primitive date-time data type called XSD: dateTime if we will not carefully use it then this will create an interoperability problem [Cohan,2002].

A Date-time is a value type object in.NET. whenever a.NET client calls a web service method DatetimeNullcheck() that contains XSD:dateTimedata type, it is easily mapped with The System.DateTime package of.NET.After mapping, the value of date type object will be stored in stack temporarily, create its own copy of data, and without change stored inside memory location [Shetty & Vadivel,2009]. Hence, whenever a null is entered in this web service method as shown in Fig 5, .NET treats null as value and generates no error condition.

But in JAVA Date-time data type is treated as a reference type. When the same web service method is called by JAVA client, the Mapping of XSD:dateTime is done with JAVA.util.Date or JAVA.unil.Calender and after mapping Date-time object will be stored in heap. But when a null is entered in this web service method, it will not be stored in heap rather will a raise Null pointer exception or unparsable exception. Because null is

treated as a value in JAVA and Date–time is a reference type object. Due to this mismatch, an exception will always arise, or we can say interoperability issue occurs at JAVA client side as shown in Fig 6.

```
[Web Method]
public DateTimegetDateTimeNullCheck(DateTime? _Strdatetime)
{
DateTime _datetime = Convert.ToDateTime(_Strdatetime);
return _datetime;
}
```

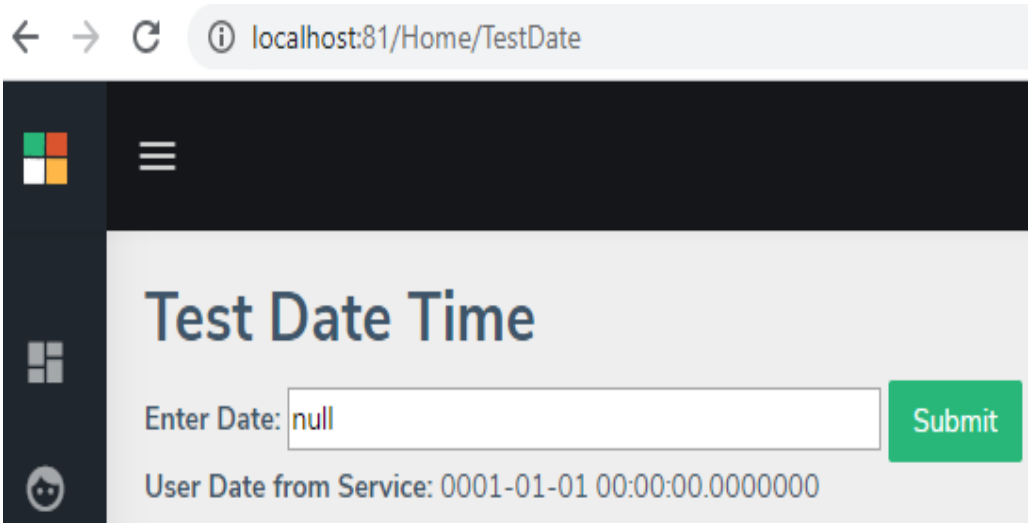

Fig. 5 Testing of null at.NET client side

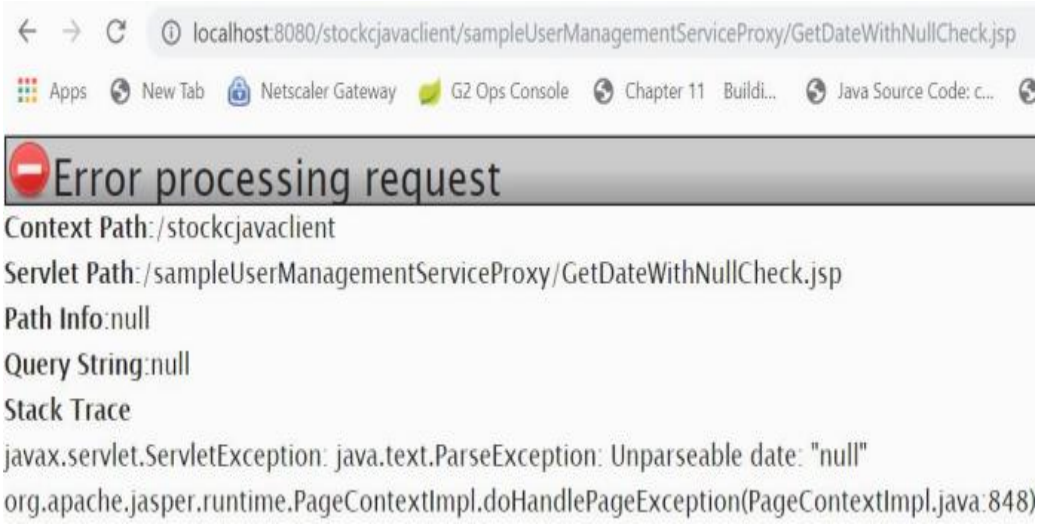

Fig. 6 Testing of null at JAVA client side

Collection of Complex Data Types

Apart from primitive data types, .NET and JAVA, both are enriched with collection type's libraries. Many collection objects are common in them like Array, stack, queue, vectors, Hash table, etc [Shetty & Vadivel, 2009]. ArrayList is the most common collection type object used by every language. These collections objects are called weekly typed objects as it holds elements of different data types [Cohen, 2002]. A web service built in.NET platform contains "ArrayList" as collection type and when this web service is converted into the WSDL file the "ArrayList" automatically changed into "ArrayOfAnyType" of XSD schema [Kumar et.al, 2003]. It means whenever a.NET client consumes this web service, it works accurately in the.NET platform.

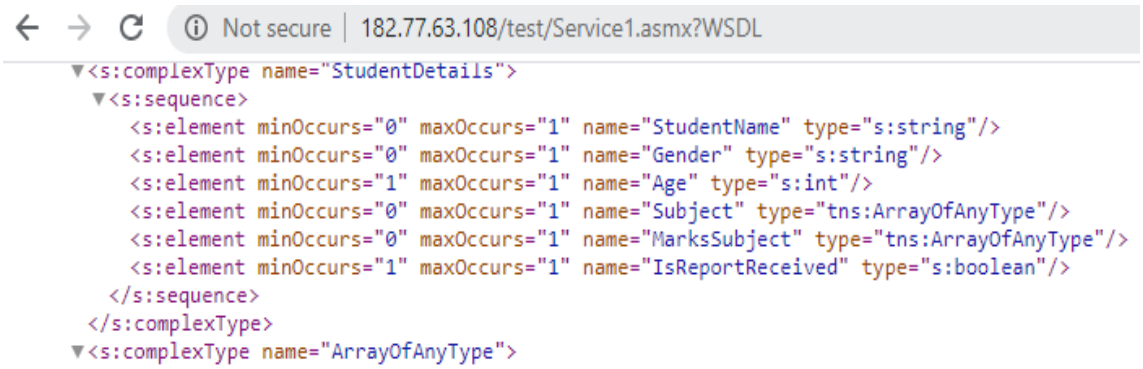

Fig. 7 WSDL file of web service that contain 'Complex type' Data.

To prove this fact, we have created a student Detail form in.NET web service (service1.asmx) that carry two methods InsertStudentDetails() and getStudentsDetails(). Both methods are using "ArrayList" as complex collection type as shown in code below and when web service converted into WSDL file, both methods changed into "ArrayOfAnyType" as shown in Fig.7 and The student data will be stored in the student.xml file as shown in Fig 8.

```
[WebMethod]
public List<StudentDetails>GetStudentDetails()
{
List<StudentDetails>lst = new List<StudentDetails>();
DataSet ds = new DataSet();
ds.ReadXml(Server.MapPath("StudentDetails.xml"));
----------------------------------------
ArrayListSubjectal = new ArrayList(SubjectArray);
ArrayListMarksSubjectal = new ArrayList(MarksSubjectArray);
}
```

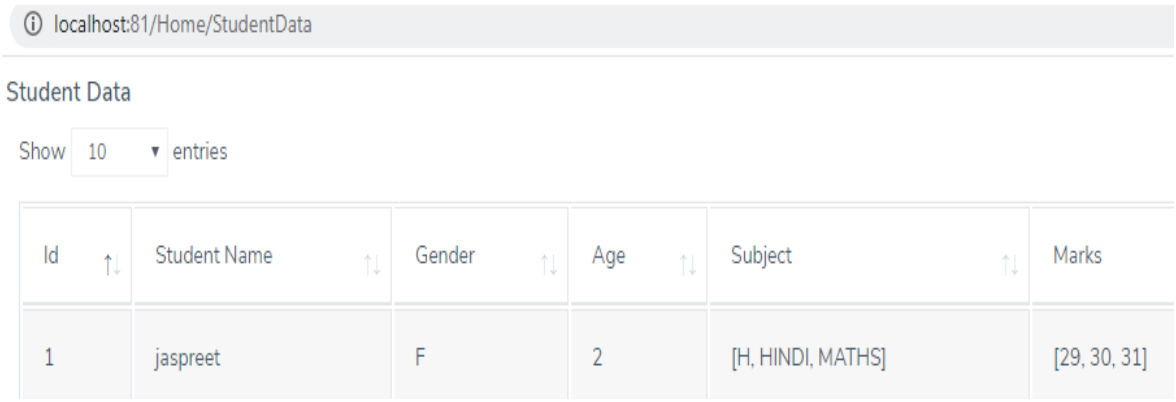

Fig. 8 StudentData.xml file stored in database of 'Students Details

The same thing is not possible at the JAVA client side. A JAVA client also contains "ArrayList" as a collection type. Whenever a JAVA client tries to consume this XML web service (WSDL), there will be a mismatch of data types issues that occur [Balmabrouk et .al, 2016]. It means JAVA does not support or automatically convert "ArrayList" into "ArrayOfAnyType" collection object. Fig 9 shows the code snapshot of JSP file "insertstudentDetails.jsp" at JAVA client-side and Fig 10 shows whenever a JAVA client tries to consume this WSDL file, a data mismatch error occurs. Hence an interoperability issue arises at the JAVA client side.

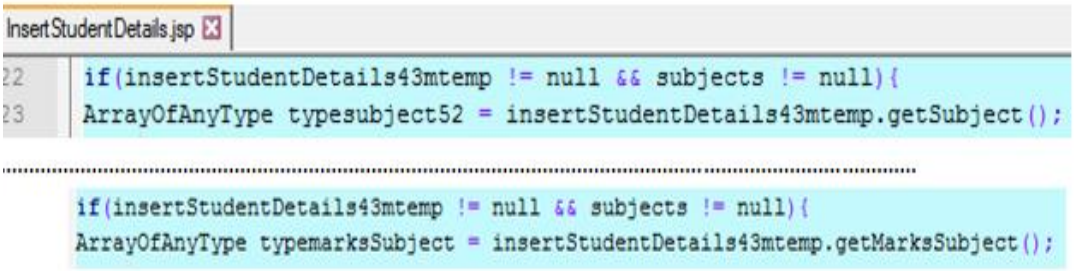

Fig. 9 "InsertStudentdetails.jsp" File at JAVA client side.

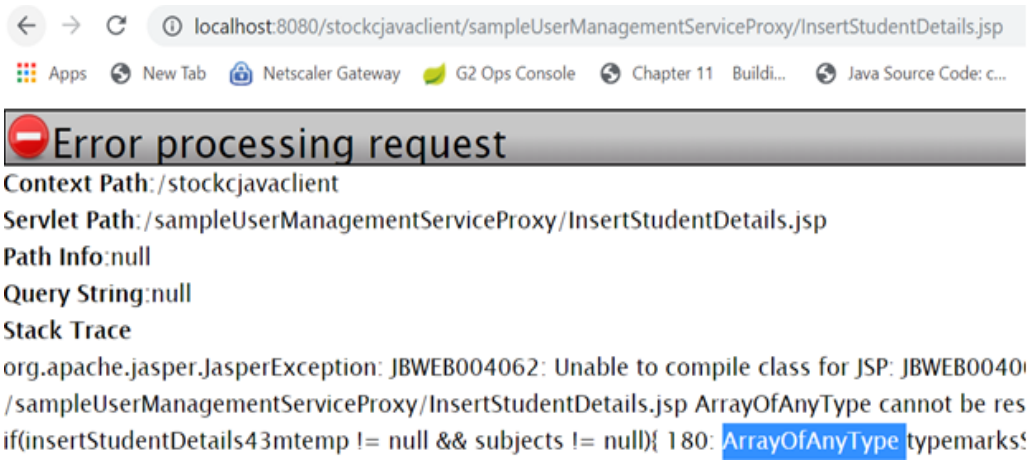

Fig. 10 Snapshot of "Data type mismatch" Error at JAVA client side

Integrated Architecture of JADE-WSIG

JADE is a software tool that connects web services with agent technology by using special add on WSIG. JADE provides interoperable, dynamic, distributed, peer to peer and, ease to use the environment to web services. As shown in Fig.11 like web service's UDDI, JADE can also publish its services to JADE-DF. It is assumed that both the environment work parallel to each other.

Moreover, features of both the technologies resemble, but the capability of building agents makes agents different from web services [Liu, 2017]. JADE includes all the libraries of JAVA classes to run the Agents and provide basic services. JADE-WSIG does the mapping between web services and a multi-agent so that both can communicate with each other [Bellifeminea, 2008]. It also does language transformation between ACL & SOAP and ACL & WSDL to pass a message between web service and agent clients [Labrou et. al, 1999].

As shown in Fig 11, for every web service, clients send a request to another client or can search in UDDI. Most of the services build in the.NET environment is easily invoked by JAVA client but in some cases, it shows interoperability issues as discussed above. In that case, UDDI sends a request to Gateway to find the exact match of service from JADE-DF [Bennajeh & Hachicha, 2015].

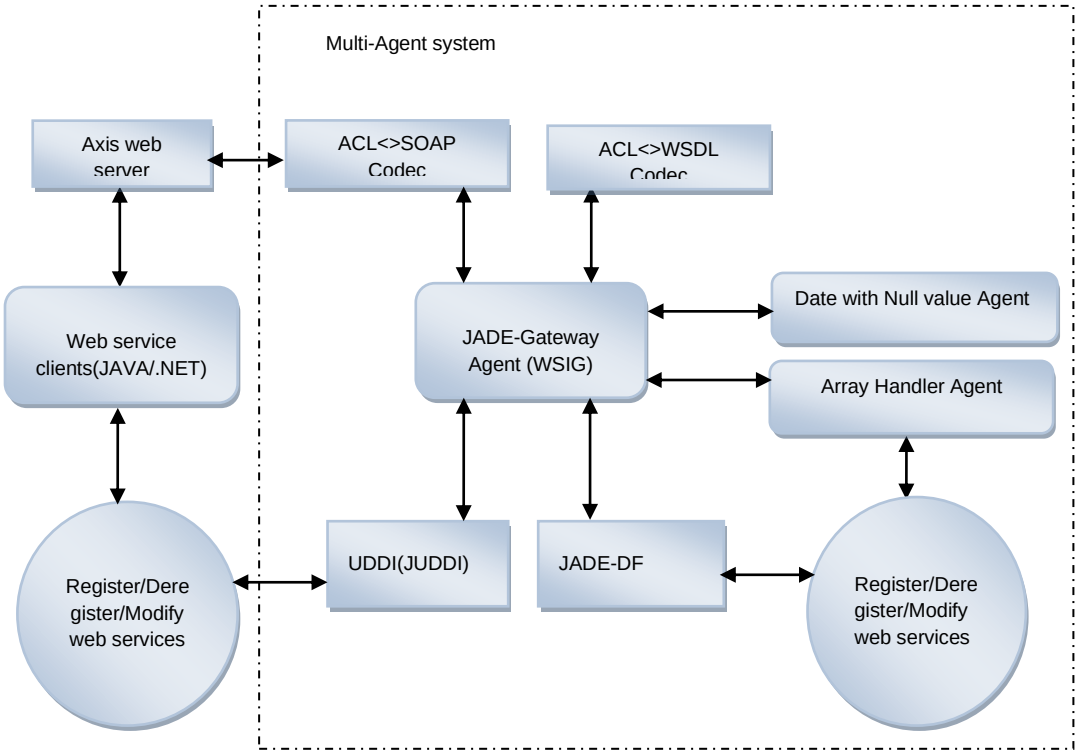

Fig. 11 An integrated Architecture of JADE-WSIG

The middleware JADE-WSIG passes the request to the Agent system, to get a match of the web service from Agent-DF. If the match found in the DF then a particular agent service is returned to gateway otherwise agent will built a new service agent as according to the properties of interoperable web service and finally, this new agent service work on the interoperable issue and after resolving the issue, it will be stored in AGENT-DF and UDDI by gateway [Bennajeh & Hachicha, 2015]. In this way, the gateway sends the matched service to UDDI or back to web service client.

To remove these issues the following tasks is to be done by JADE-WSIG [Chawla et al., 2019].

1. As both the technology working on the same set of specifications, so JADE-WSIG acts as a bridge between them for sending and receiving requests and responses [Greenwood et al., 2007].
2. JADE-WSIG uses service discovery operation to finds a match of web service in agent-DF and do the service description transformation and vice-versa. Every web and agent service has a unique service id. The search for service from a particular directory is possible by clients or JADE only with their ids [Bellifeminea et al., 2008].
3. The service description transformation covert WSDL to DF Agent description to publish service in Agent-DF and in the same way when an agent is required, DF agent description converts to WSDL to publish a service in UDDI.
4. Once the service is published in either agent-DF or UDDI, a service invocation is required for protocol conversion. It converts ACL<>SOAP and ACL<>WSDL codec to communicate the web service to agent clients and vice versa [Nguyen et al, 2007].
5. JADE includes the AMS (agent Management system) to coordinate and execute the agent services. AMS helps the agent to connect with the remote agent platform or modify its description according to the private agent address.
6. To do the whole process, we have created two agents with the help of JADE-WSIG to resolve two different interoperability issues. Date with null value Agent will resolve the null value issue and the Array handler agent will resolve the data type error "ArrayOfAnyType" of JAVA client.
7. Gateway provides a set of classes for ontology mapping and graphical tools for monitoring and analyzing the running agents [Nguyen et al, 2007].

JADE-Results

On the official website of JADE number of documents and tutorials are available to easily download the updated version of JADE.A graphical toolkit supported by JADE also

provide support to development, testing, debugging and monitoring phase to the software applications. The DF agent of JADE itself has its own GUI as shown in Fig 13.A jade system contains thousands of agents for transferring a large amount of messages. In this paper, we have used JADE with WSIG to resolve two interoperability issues as discussed above. With the help of JADE, we have created two agents one to handle Date with null value and other to handle the collection of complex data types as shown in Fig. 12.

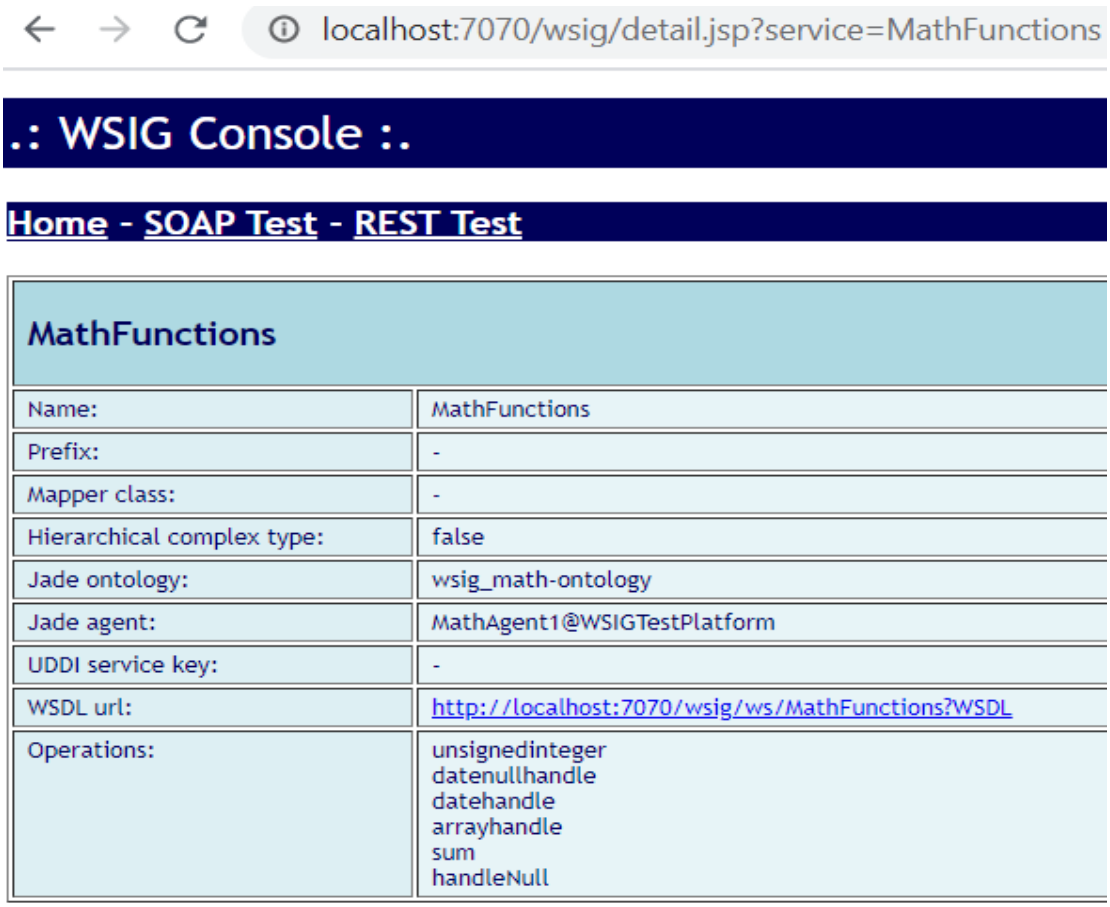

Fig. 12 Building of Agents using JADE-WSIG environment.

The experimental setup configuration for resolving web service issues are shown in Table 1.The proposed architecture is implemented with processor Intel core i5-6200U with 4GB RAM and window10, and 64 bit operating system. During message sending and calling, the client application uses RPC-SOAP Protocols. JBOSS-wildfly 8.x is used to handle JAVA web services that are embedded in JSP files using the eclipse-IDE software version 4.15.microsoft.NET IIS Server 10.0 is used to handle.NET web services that client call from WSDL. Apache Axis-2 server is used for handling clients request, fetching the services from directory and giving responses back to clients. Third part tools JADE with add-on of WSIG version 8.5.13 is used to resolve the issues. This large experimental

setup highlighting the critical interoperability issues and finally demonstrates the resolution also.

Table 1 Client side Software configuration for establishing web Service and Multi-agent system

Platform	**Server**	**Version**
JAVA	JBoss-Wildfly 8.x	8.x
.NET(C#&ASP.NET)	Microsoft-IIS	10.0
JADE	Apache-Tomcat	8.5.13
WSIG(Add-on)	Apache-Tomcat	8.5.13
My-sql command line client	My-sql Server	4.12.0

To deploy an agent task, it should define behavioral classes to add behavioral objects. An agent.setp() method is used to initialize the life cycle of agent and agent.action() help to execute the agents from ready queue. To implement the web service with agents, First WSIG has to be registered as a special agent in JADE-DF and a web service endpoint in JUDDI directory [Greenwood et al., 2004]. With the help of ontology, the data and its structure are passed from WSDL to agent services ontology translator to convert input, output, data types, and methods of web service to agent ontology. Below is the piece of ontology code build on JADE side while mapping of web service with agents.

```
import JADE.content.onto.BasicOntology;

import JADE.content.schema.AgentActionSchema;

AgentActionSchema as = (AgentActionSchema) getSchema(DATENULLHANDLE);

as.add(FIRST_ELEMENT,(PrimitiveSchema) getSchema(BasicOntology.STRING));

as.setResult((TermSchema) getSchema(BasicOntology.STRING));


AgentActionSchemaarrayHandle = (AgentActionSchema)
getSchema(ARRAYHANDLE);

arrayHandle.add(FIRST_ELEMENT, (PrimitiveSchema)
getSchema(BasicOntology.STRING));

arrayHandle.setResult((TermSchema) getSchema(BasicOntology.STRING));
```

Date Null Agent

After the mapping has been done between two technologies, the agent has to register into JADE-DF. As shown in Fig13 JADE builds a "DateNullAgent" and registered into JADE-DF. This agent work on operation datenullhandle() as shown in Fig 12, to handle the interoperability issue of date with null value at the JAVA client side.

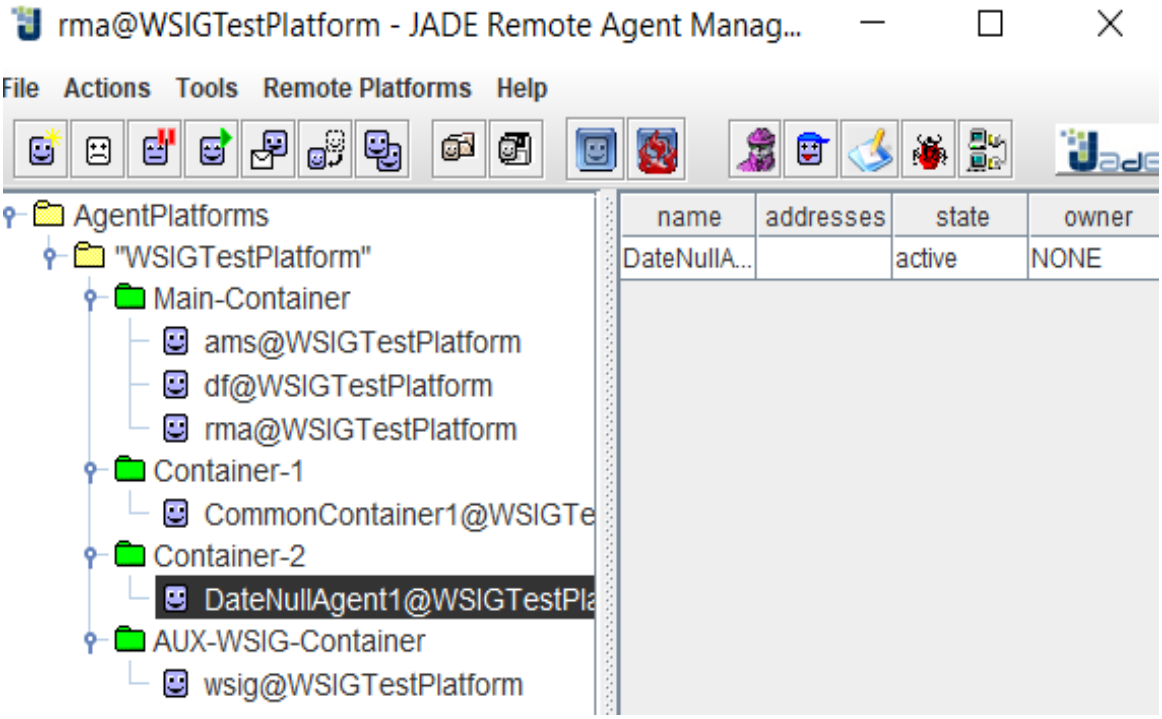

Fig. 13 "DateNullAgent" Registered in JADE-DF

DateNullAgent works on behalf of WSDL. As the JAVA client tries to fetch the WSDL of Datewithnullvalue() function from UDDI, the UDDI redirects it to JADE-WSIG. This gateway accepts the request of UDDI and calls the service discovery method to search the corresponding service from JADE-DF. After searching, it matches the description of both the services and does the language conversion. In our implementation, we have built the agents, as the agent gets a request of Date with null value method from a gateway; it replaces the null with the empty string. JADE Agent supports the whole JAVA libraries and interfaces and implemented in JAVA language [Nguyen et .al, 2005]. In addition to that, it has the capability to create its own functional method with the help of other agents. JAVA support empty string in Date-time format but does not support null. The replacement of null with empty string by JADE, make convenient for JAVA client to run the WSDL without parsable error. As shown in Fig 14, JADE client created an agent service for JAVA client with the help of WSIG. After resolving the issue, JADE-WSIG

saves a copy of every new agent to DF as well as to UDDI for further communications of services.

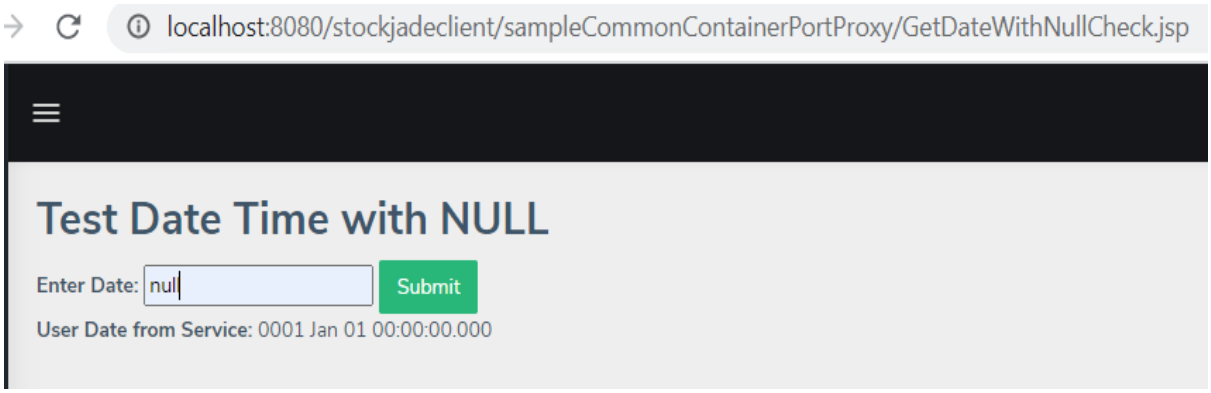

Fig. 14 Resolution of Date with Null check at JADE client side

Array Handle Agent

For the resolution of the second issue collection of complex data types, we have created a new agent called Array handle agent. In this issue, the JAVA client was facing the interoperability issue "ArrayOfAnyType" at the client-side. Whenever was JAVA client trying to fetch a WSDL that contains "ArrayOfAnyType", it was facing the data mismatch error. To resolve this issue, UDDI redirects the JAVA client and forwards the WSDL request of complex data type towards Gateway. Now gateway passed it to JADE client. If the service will be available in DF then it passed to a web service client otherwise JADE will create a new agent to resolve the service issue.

```
Private void serveArrayAction(ArrayHandlearrayHandle, Action actExpr,
ACLMessagemsg) throws RemoteException {

        String firstElement= arrayHandle.getFirstElement();

        if(firstElement.equals("ArrayOfAnyType")) {
                firstElement ="Object[]";
        }
```

As in this case, Array handle agent replace the "ArrayOfAnyType" with an Array of objects as shown in the code. Like another array, it also store value like integer, string, Boolean, etc and location of reference variable of objects [Shetty & Vadivel, 2009 & Hamid et al., 2006]. Object array works well in the JAVA platform. In this way, the

Agent helps to convert ambiguous WSDL into valid WSDL and save a copy of new service in both DF and UDDI. As shown in Fig 15. Now the student form is filled well at the JAVA client-side by using the JADE services at the backend. The marks and subjects attributes were earlier construct in "ArrayOfAnyType" and now it is using an array of objects.

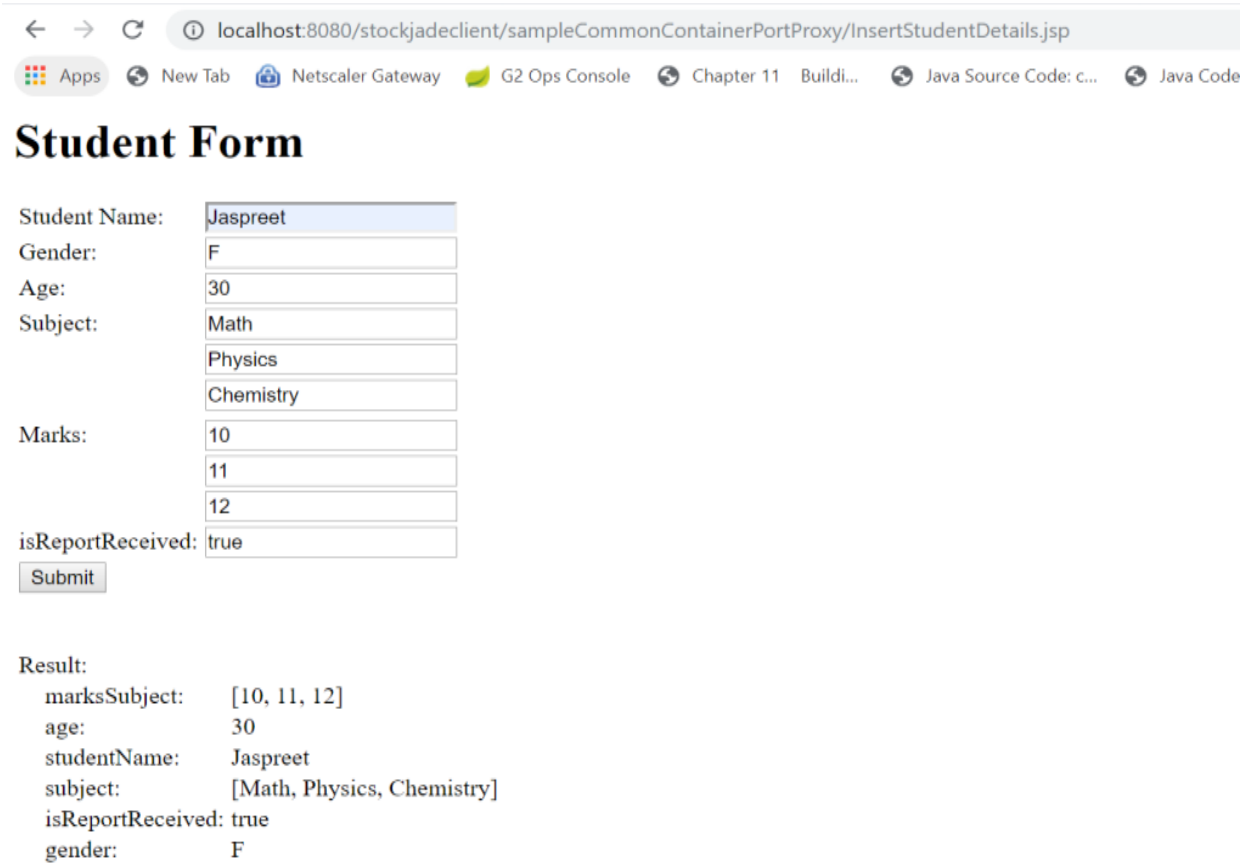

Fig. 15 Resolution of "Complex data types" at JADE client side

Discussion and Comparison of Research Work with Previous Related Work

Web service is a rich source while executing different client platforms. Different web services help in a different way and meet a milestone at the client side. The standard approach help to achieve syntax interoperability but to meet with symmetric and structural level interoperability is a complex and crucial task. Many Authors has been discussed about agent technology and used different tools and technologies to remove interoperable issues. Few of tools are Agent development kit, FIPA-OS, Grasshoppers, JACK intelligent agents, JADE, LEAP, ZEUS, java agent services API, Aglets, Anchors and voyager, cougar, 3APL.Table 2 shows the comparisons of major five Agent technologies with list of parameters.

Table 2 Comparisons of various Toolkits

Toolkit / Parameters	JADE	IBM-Aglet	voyager	Anchor	Zeus
Nature of product	Open source tool	Open source tool	Commercial tool	Licensed tool	Open source tool
Communication Mechanism	Asynchronous	Synchronous and Asynchronous	Synchronous	Asynchronous	Asynchronous
Interoperability	Yes, compliance with FIPA Standards	Mobile agent system interoperability facilities(MASIF) but not interoperable to other systems	Restricted access for clients/servers	On limited platforms	No
Security Mechanism	Good	weak	weak	strong	Good
Migration Mechanism	RMI	SOCKET	RMI	SOCKET	NIL
usability	Business firms, researcher labs and centers	Electronic market	Built GUI of java applications	Commercial and medical industry	Commercial and intelligent systems
Agent-Mobility	A little weak	weak	weak	weak	nil
GUI	YES	Partially	NO	yes	YES
Message passing	FIPA(YES)	YES	YES	NO	YES(FIPA)
Level of activities	Very high	some	low	no	no

Conclusion

Sometimes semantic level interoperability issues become a big challenge while executing the web service. This paper has described how web services are using envisioned capabilities of the multi-agent system to resolve the interoperability and how the agent-based web service building the whole system unambiguous and robust. JADE-WSIG acts as a gateway and helps to service invocation from web service clients. The main objective of JADE-WSIG was to invoke the web service from UDDI, discover the web service in JADE-DF, match the service description of agent service with web service, do the ontology mapping and language conversion and if required build a new agent. In this paper, we have worked on two interoperability issues of web service. We have shown by the implementation that how interoperability issues occur at JAVA client-side while fetching the WSDL files meawhile.NET client was working well with the same WSDL. Finally, to resolve the issues, we have been established a set up of JADE with WSIG and shows how interoperability issues of JAVA clients are resolved by JADE client. So in this way we have tried to resolve the interoperability issues with the help of JADE-WSIG.

References

Ibrahim, N.M., & Hassan, M.F. (2019). A comprehensive comparative study of MOM for adaptive interoperability communications in service oriented architecture. *International Journal of Trend in Scientific Research and Development, 3*(3), 23-30.

Chawla, J., Ahlawat, A.K., & Goswami, G. (2018). A review on web services interoperability issues. *In 5th IEEE Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON)*, 1-5.

Chawla, J., & Goswami, G. (2019). Integrated Architecture of Web Services using Multiagent System for Minimizing Interoperability. *In IEEE 6th International Conference on Computing for Sustainable Global Development (INDIACom)*, 1062-1067.

Kanhere, D.A., & Raja, S.J. (2018). Multi-Agent Systems. *A survey. IEEE Access*, *6*, 28573-28593.

Kamdar, R., Paliwal, P., & Kumar, Y. (2018). A state of art review on various aspects of multi-agent system. *Journal of Circuits, Systems and Computers*, *27*(11), 1830006.

Haile, N., & Altmann, J. (2018). Evaluating investments in portability and interoperability between software service platforms. *Future Generation Computer Systems*, *78*, 224-241.

Jing X., & Chen-ching L. (2017). Multi-agent systems and their applications. *Journal of International Council on Electrical Engineering*, *7*(1), 188–197.

Karima, B., Fatima, B., & Maroua, B. (2016). Multiagent based Model for web service composition, *International Journal of Advance computer science and applications*, *7*(3), 144-150.

Anouer, B., Hela, H. (2015). Web Service Composition Based on a Multi-agent System. *Conference: the 4th Computer Science On-line Conference, 3*, 295-305.

Coria, J.A.G., Castellanos-Garzón, J.A., & Corchado, J.M. (2014). Intelligent business processes composition based on multi-agent systems. *Expert Systems with Applications*, *41*(4), 1189-1205.

Elia, I.A., Laranjeiro, N., & Vieira, M. (2014). Understanding interoperability issues of web service frameworks. *In 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, 323-330.

Johari, K., & Kaur, A. (2013). Some Interoperability Issues in The Designing of Web Services: Case Study on Credit Card. *International Journal on Web Service Computing*, *4*(4), 11-20.

Udayakumar, R., Thooyamani, K.P., & Khanaa, V. (2003). A Comparison of J2EE and.NET as Platforms for Developing E-Government Applications. *International Journal of Engineering Research and Development*, *7*(1), 116-121.

Poggi, A., & Tomaiuolo, M. (2010). Extending the JADE Framework for Semantic Peer-To-Peer Service Based Applications. *In Developing Advanced Web Services through P2P Computing and Autonomous Agents: Trends and Innovations*, IGI Global, 18-35.

Shetty, S., & Vadivel, S. (2009). Interoperability issues seen in Web Services. *IJCSNS International Journal of Computer Science and Network Security*, *9*(8), 160-169.

Kim, B., Son, B., Han, S., & Youn, H. (2009). Agent platform-based directory facilitator for efficient service discovery. *In IEEE International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, 101-107.

Bellifemine, F., Caire, G., Poggi, A., & Rimassa, G. (2008). JADE: A software framework for developing multi-agent applications. Lessons learned. *Information and Software Technology*, *50*(1-2), 10-21.

Greenwood, D., Lyell, M., Mallya, A., & Suguri, H. (2007). The IEEE FIPA approach to integrating software agents and web services. *In Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems* 1-7.

Nguyen, X.T., & Kowalczyk, R. (2007). Ws2jade: Integrating web service with jade agents. In *International Workshop on Service-Oriented Computing: Agents, Semantics, and Engineering*, 147-159.

Nezhad, H.R.M., Benatallah, B., Casati, F., & Toumani, F. (2006). Web services interoperability specifications. *Computer*, *39*(5), 24-32.

Nguyen, X.T., & Kowalczyk, R. (2005). Enabling agent-based management of Web services with WS2JADE. *In Fifth International Conference on Quality Software (QSIC'05)*, 407-412.

Greenwood, D., & Calisti, M. (2004). Engineering web service-agent integration. *In IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No. 04CH37583)*, 2, 1918-1925.

Huhns, M.N. (2002). Software agents: the future of web services. In *Net. ObjectDays: International Conference on Object-Oriented and Internet-Based Technologies, Concepts, and Applications for a Networked World*, 1-18.

Labrou, Y., Finin, T., & Peng, Y. (1999). Agent communication languages: The current landscape. *IEEE Intelligent Systems and Their Applications*, *14*(2), 45-52.

Frank, C. (2002). *Understanding web service interoperability*, IBM-Developer work.

Nwana, H.S., & Ndumu, D.T. (1998). A brief introduction to software agent technology. *In Agent technology*, 29-47.

Nwana, H.S. (1996). Software agents: An overview. *Knowledge engineering review*, *11*(3), 205-244.

Siddig, J.M. (2015). *Comparison and evaluation of performance of mobile agent toolkits (JADE and Aglet)* (Doctoral dissertation, Sudan university of scince and technology).

Bergenti, F., Caire, G., Monica, S., & Poggi, A. (2020). The first twenty years of agent-based software development with JADE. *Autonomous Agents and Multi-Agent Systems*, *34*, 1-19.

Petrosino, G., & Bergenti, F. (2019). Extending Message Handlers with Pattern Matching in the Jadescript Programming Language. *In WOA*,113-118.

Aarti, S., Dimple, J., & Sharma, A.K. (2011). Agent Development Toolkits. *International Journal of Advancements in Technology*, *1*, 158-165.

http://www.ws-i.org/Docs/ws-i_faq_01.pdf

https://www.ibm.com/developerworks/webservices/tutorials/ws-understand-web-services6/index.html

http://www.fipa.org

https://jade.tilab.com/papers/2005/JADEWorkshopAAMAS/AAMAS05_JADE-Tutorial_WSIG-Slides.pdf

https://jade.tilab.com/