JAX-WS Web Service for Transferring Image

Girish M. Tere
Department of Computer Science,
Shivaji University,
Kolhapur,
Maharashtra – 416004, India
girish.tere@gmail.com

R. R. Mudholkar
Department of Electronics,
Shivaji University,
Kolhapur,
Maharashtra – 416004, India

B. T. Jadhav
Department of Electronics and Computer Science,
Y.C. Institute of Science,
Satara, Maharashtra - 415001, India

***Abstract*—Using Web Services, it is possible to send any type of information in any form of encryption. In this context, different techniques have been used to attach binary files in SOAP messages. Web services provide communication between different platforms by using standardized SOAP message protocols. Hence, it is often productive to implement web service interfaces for the integration of experimental infrastructure into a collaborative framework. Transfer and management of experimental data is an important problem. SOAP messaging is primarily oriented towards XML character data, with binary data transmitted with, for example, a base64 character encoding. However, as scientific experiments often generate binary data, transferring binary information by applying an encoding algorithm can slow down the performance of the system. The notion of web service attachment has been introduced to solve this problem. In this paper, we illustrate how we use web services with attachments to improve binary data transfer performance. This paper discusses complete example of how to use MTOM in JAX-WS.**

Keywords— base64Binary, MTOM, SOAP, XOP

I. INTRODUCTION

In heterogeneous environment communication between client application and Web service is possible by exchanging XML documents as shown in Figure 1.

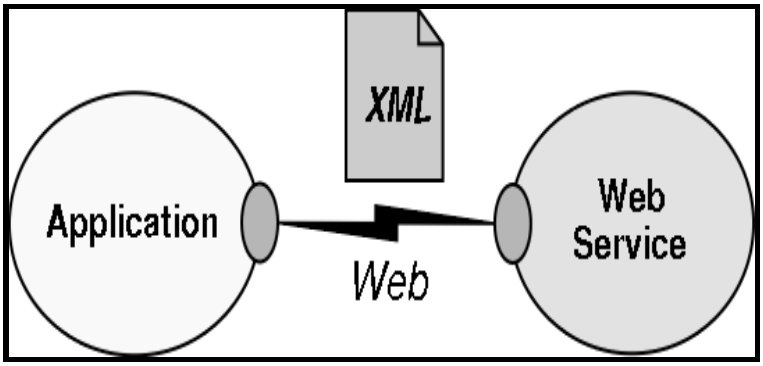

Figure 1. Communication between Client and Web service

MTOM [6] is a standard which allows Web services to transfer binary data efficiently and conveniently. SOAP Message Transmission Optimization Mechanism/XML-binary Optimized Packaging (MTOM/XOP) defines a method for optimizing the transmission of XML data of type xs:base64Binary [2] in SOAP messages. When the transport protocol is HTTP, MIME attachments are used to carry that data while at the same time allowing both the sender and the receiver direct access to the XML data in the SOAP message without having to

be aware that any MIME artifacts were used to marshal the base64Binary data. The binary data optimization process involves the following steps:

1) encode the binary data

2) remove the binary data from the SOAP envelope

3) compress the binary data

4) attach the binary data to the MIME package

5) add references to the MIME package in the SOAP envelope

Message Transmission Optimization Mechanism (MTOM) provides a way to send binary data to Web Services in standard SOAP messages [7]. MTOM leverages the include mechanism defined by XML Optimized Packaging (XOP) whereby binary data can be sent as a MIME attachment (similar to SOAP with Attachments) to a SOAP message. The binary data can then be referenced in the SOAP message using the <xop:Include> element. XOP is an alternate serialization of XML that just happens to look like a MIME multipart/related package, with an XML document as the root part. That root part is very similar to the XML serialization of the document, except that base64-encoded [9], [13] data is replaced by a reference to one of the MIME parts, which isn't base64 encoded.

This allows us to avoid the bulk and overhead in processing associated with encoding, the only way that one can fit binary data directly into an XML world. XOP can be used for any XML-based format; MTOM is just a description of how XOP is layered into the SOAP HTTP transport.

II. BINARY EXCHANGE WITH MESSAGE TRANSMISSION OPTIMIZATION MECHANISM (MTOM)

Basic Web service architecture is shown in Figure 2. For any Web service, communication between Service provider, Service Broker and Service consumer is done by exchanging SOAP documents written in XML. In spite of the flexibility and acceptance of XML in application architecture, there are certain limitations associated with XML.

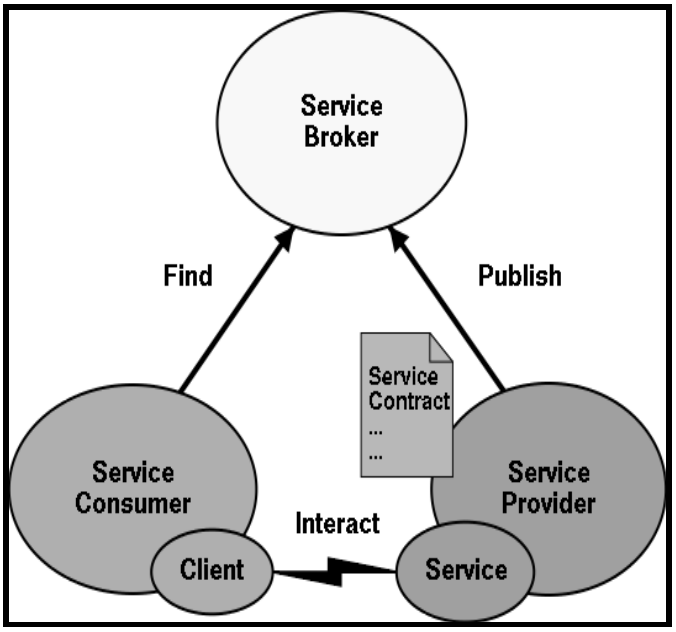

Figure 2. Architecture of Web service

These limitations are for example media data and data that includes digital signatures are two of the most common examples. The reason is with media data there are highly standardized formats where compression plays a major part. While for digital signatures, the binary integrity required would not be preserved if it were serialized into a structured XML format. At the same time serializing binary data proved to be extremely resource intensive. As a result there are various methods for sending binary data through the use of Web services. These are:

* XML Representation: Replace any non-Xml data with a structured XML representation and include it in our message. The problem here is that it bloats the data tremendously. However, if we are sending something like Microsoft Word documents, we might want to consider saving them in XML format.

* SOAP with Attachments (SwA): Use SOAP with Attachments to send binary data with SOAP in a MIME Multipart message. SOAP with Attachments was the first attempt by Microsoft (along with Hewlett Packard) at solving the opaque data/attachments problem. Unfortunately SwA had poor composability and it did not work well with the WS-* specifications.

* WS-Attachments with DIME: Compared to SOAP with attachments this is a faster and more efficient solution to sending binary data. However it also has the same composability issues as SwA. As in the case with SwA, it does not use a SOAP envelope approach to messaging. This limits DIME's ability of the higher-level Web service specifications to properly work upon a message.

* Base 64 Encoding: This method of encoding has been used for quite some time and is well known for its binary data problem. Although base-64-encoded data is usually more efficient from a message-size perspective than using XML Representation, it still increases the data size by about 33% even when only single byte characters are used.

* Message Transmission and Optimization Mechanism (MTOM): MTOM is widely accepted today in binary transport. The MTOM approach take advantage of the SOAP infrastructure, but gain the transport efficiencies provided by SOAP with Attachments solution [12]. MTOM implementation combines the benefits of composability of Base 64 encoding with the transport efficiency of SwA. The binary data is dealt with just like it is with SwA it is simply streamed as binary data as one of the Multipurpose Internet Mail Extensions (MIME) message parts.

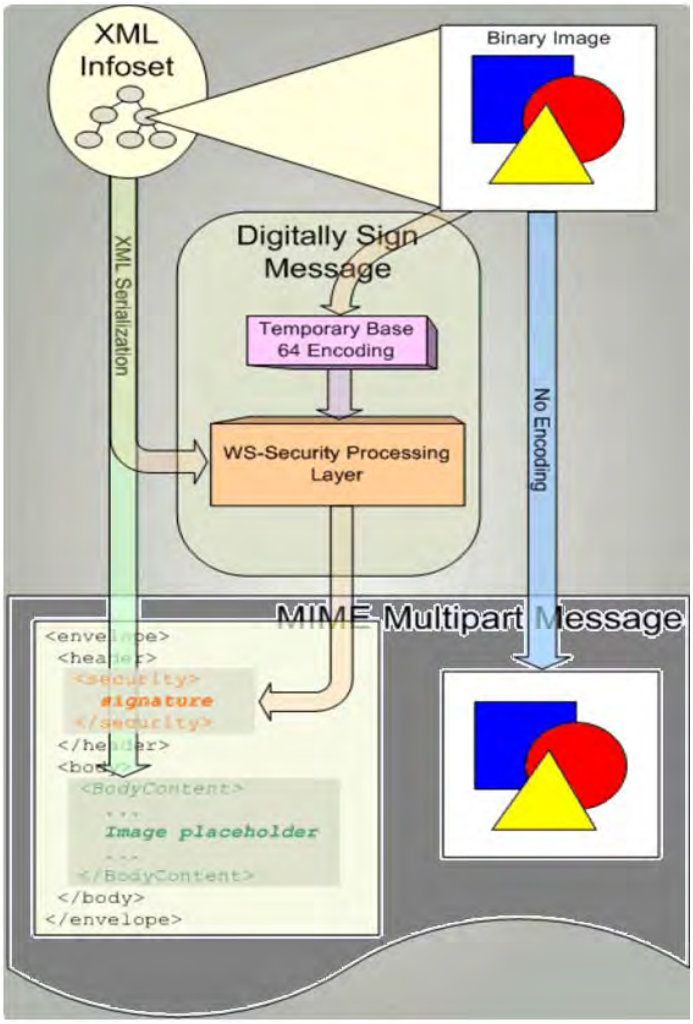

Figure 3. MTOM working

The binary data is streamed through a layer in which it is temporarily encoded to base-64. During the streaming the signature of the message is calculated. The encoded base-64 data is not transferred or held in memory. On deserialization, the same concepts apply. The raw binary is transferred directly to the higher layer, but any processing layers that access it would do so through a temporary base-64-encoding layer. This is explained in Figure 3 above.

MTOM (Message Transmission Optimization Mechanism) is the new specification for optimizing the transmission and/or wire format of SOAP messages. Primarily this means that we have a new standard that allows the sending of attachments over Web Services - one that the industry agrees on, and one that is composable with the other WS-* specifications.

The technologies of Message Transmission Optimization Mechanism (MTOM) and XML-Binary Optimized Packaging (XOP) are concerned with the fundamental issue of binary data transmission, which is of great importance to web service performance. Since becoming W3C standards in 2005, XOP and MTOM have been quickly and widely adopted in next-generation SOAP engines. In hindsight, one has to wonder why they were not proposed earlier in the development of web services or as part of the original SOAP specification. Table 1 summarizes the steps required to use MTOM/XOP to send base64Binary attachments [6].

TABLE I. STEPS TO USE MTOM/XOP TO SEND BINARY DATA [6]

No	Step	Description
1	Annotate the data types that is to be used as an MTOM attachment. (Optional)	Depending on programming model, one can annotate Java class or WSDL to define the content types that are used for sending binary data. This step is optional. By default, XML binary types are mapped to Java byte[].
2	Enable MTOM on the Web Service.	javax.xml.ws.soap.MTO
3	Enable MTOM on the client of the Web Service.	*//codes enable MTOM in client* BindingProvider bp = (BindingProvider) imageServer; SOAPBinding binding = (SOAPBinding) bp.getBinding(); binding.setMTOMEnabled(true);
4	Set the attachment threshold.	Set the attachment threshold to specify when the xs:binary64 data is sent inline or as an attachment.

III. EXPERIMENT

We developed a web service using JDK 1.6 and Eclipse as IDE is used for development.

Our testbed consists of following computers and router as shown in Figure 4.

- Asus, Eee PC 1015P Seashell series, Intel Atom with 1 GB RAM and Windows XP. This is used as client
- D-Link DIR-300 Wireless G Router
- DELL INSPIRON with Intel Core2 Duo CPU T6400 @ 2 GHz and 4 GB RAM with Windows XP. This is used for publishing Web service.

Figure 4. Testbed

Following steps are performed to transfer image using MTOM techniques

- Enabling MTOM on server
  Enable server to send attachment via MTOM is easy, just annotate the web service implementation class with javax.xml.ws.soap.MTOM.
- WebService Endpoint
  Initially we used RPC-style web service, published two methods, downloadImage(String name) and uploadImage(Image data), to let user upload or download an image file.
- Develop ImageServerImpl.java
- Develop ImagePublisher.java
- WebService Client: ImageClient.java

Figure 5 shows a web service client, to access the published web service at URL "http:/192.168.0.103:9999/ws/image".

192.168.0.103:9999/ws/ima

192.168.0.103:9999/ws/image?wsdl

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<!--
    Published by JAX-WS RI at http://jax-ws.dev.java.net. RI's version is JAX-WS RI 2.1.6 in JDK 6.
  -->
<!--
    Generated by JAX-WS RI at http://jax-ws.dev.java.net. RI's version is JAX-WS RI 2.1.6 in JDK 6.
  -->
<definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tns="http://ws.girish.com/" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns="http://schemas.xmlsoap.org/wsdl/" targetNamespace="http://ws.girish.com/" name="ImageServerImplService">
  <types/>
  <message name="downloadImage">
    <part name="arg0" type="xsd:string"/>
  </message>
  <message name="downloadImageResponse">
    <part name="return" type="xsd:base64Binary"/>
  </message>
  <message name="uploadImage">
    <part name="arg0" type="xsd:base64Binary"/>
  </message>
  <message name="uploadImageResponse">
    <part name="return" type="xsd:string"/>
  </message>
  <portType name="ImageServer">
    <operation name="downloadImage">
      <input message="tns:downloadImage"/>
      <output message="tns:downloadImageResponse"/>
    </operation>
    <operation name="uploadImage">
      <input message="tns:uploadImage"/>
      <output message="tns:uploadImageResponse"/>
    </operation>
  </portType>
  <binding name="ImageServerImplPortBinding" type="tns:ImageServer">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="rpc"/>
    <operation name="downloadImage">
      <soap:operation soapAction=""/>
      <input>
        <soap:body use="literal" namespace="http://ws.girish.com/"/>
      </input>
      <output>
```

Figure 5. WSDL for Image transfer web service

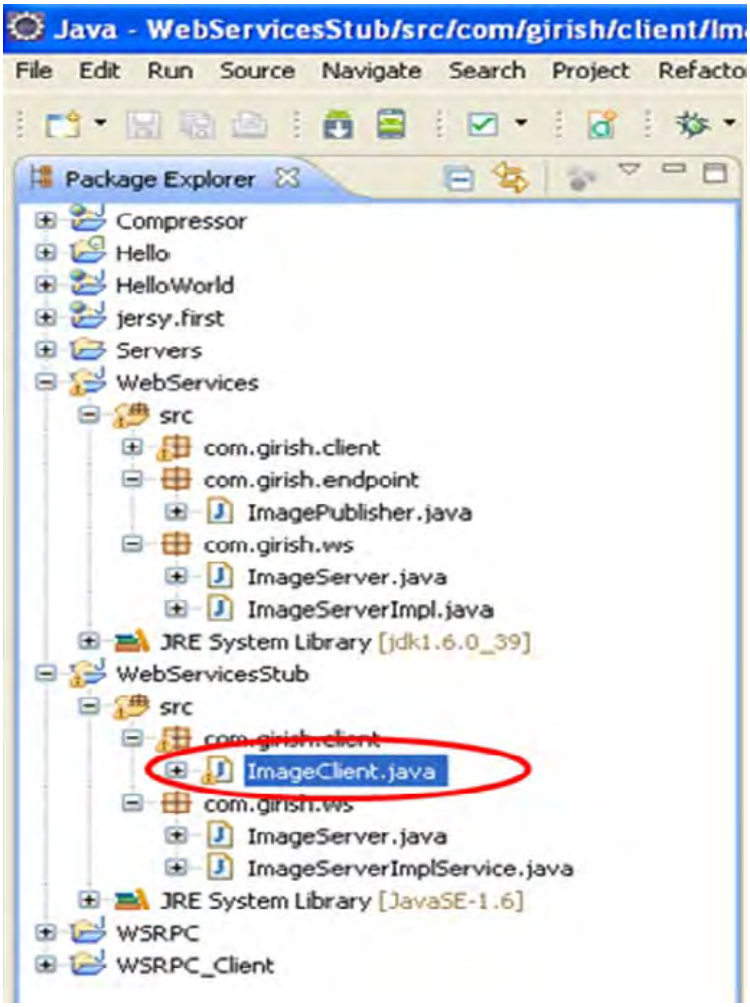

Figure 6. WebServices and WebServicesStub projects in Eclipse

A. Downloading Image at Client

Downloaded image is displayed in a frame. Screen shot is shown in Figure 7.

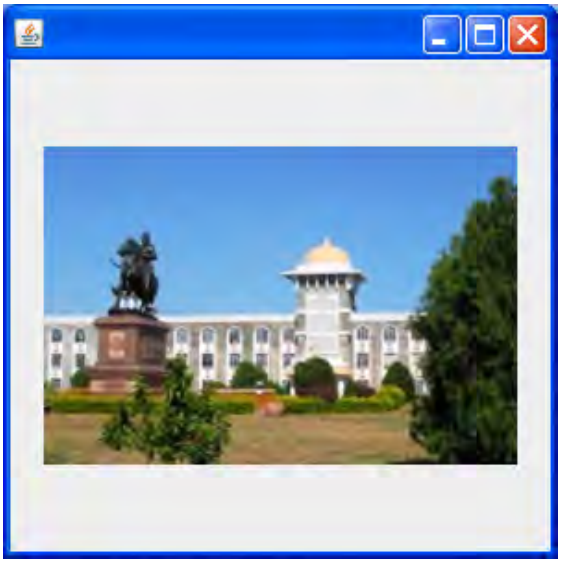

Figure 7. Downloaded image at client side

B. Uploading Image to server

After successful image upload operation output is shown in Figure 8.

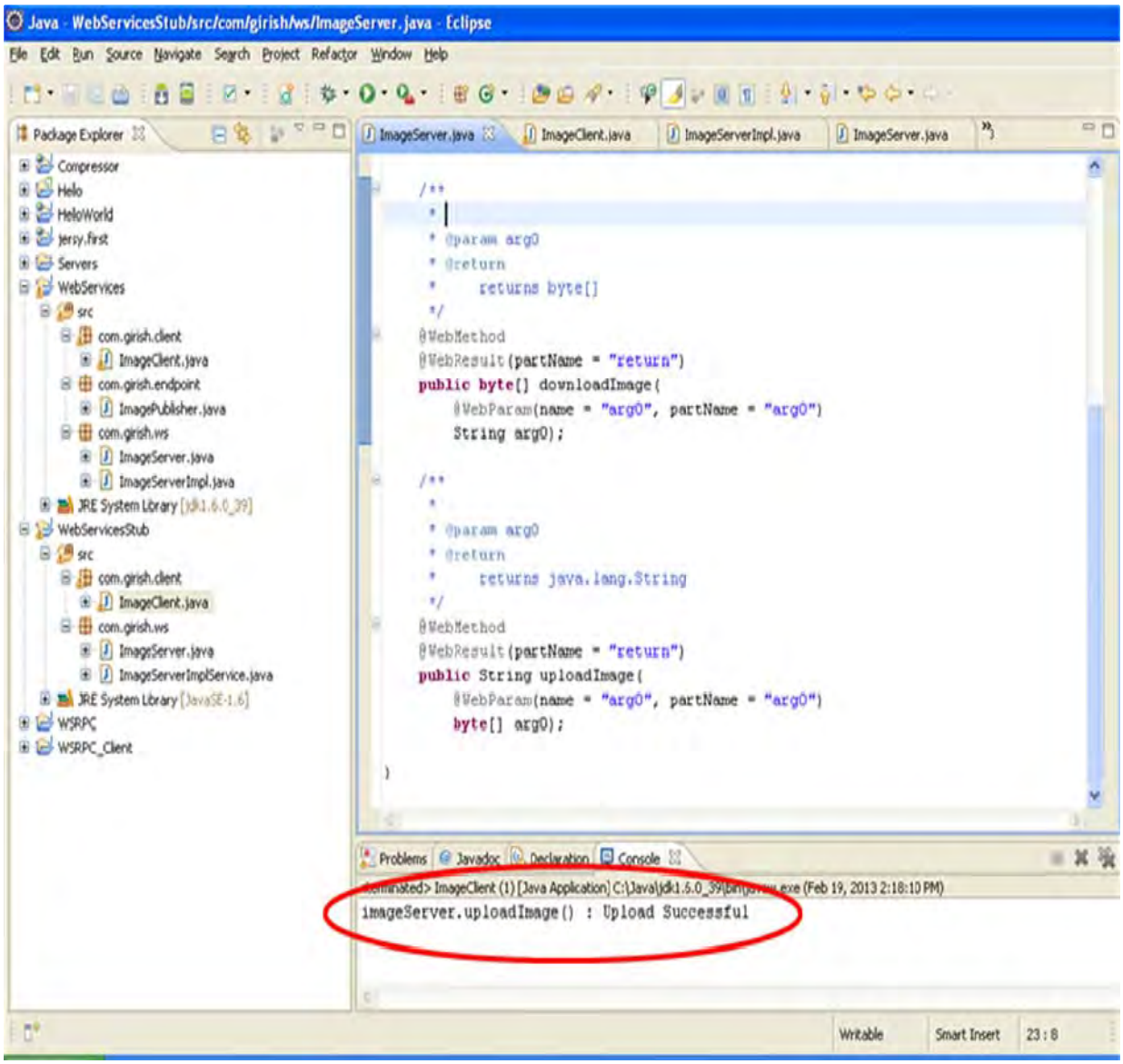

Figure 8. Uploaded Image at Server side

IV. PERFORMANCE MEASUREMENT

We measured time required for completing download and upload operation of web service in different networks. Web service is developed using RPC style (discussed in this paper) as well as in Document style. Data collected is shown in Table II and graphically in Figure 9.

Table II. TIME NEEDED (IN MSEC*) FOR DOWNLOAD/UPLOAD IMAGE IN DIFFERENT NETWORKS

	Wi-Fi		Bluetooth		Wired	
	Download	Upload	Download	Upload	Download	Upload
JAX-WS, RPC	435	523	684	756	356	376
JAX-WS, Document	387	406	624	672	287	312

*We measured response time using System.milliSeconds() method available in java.util package.

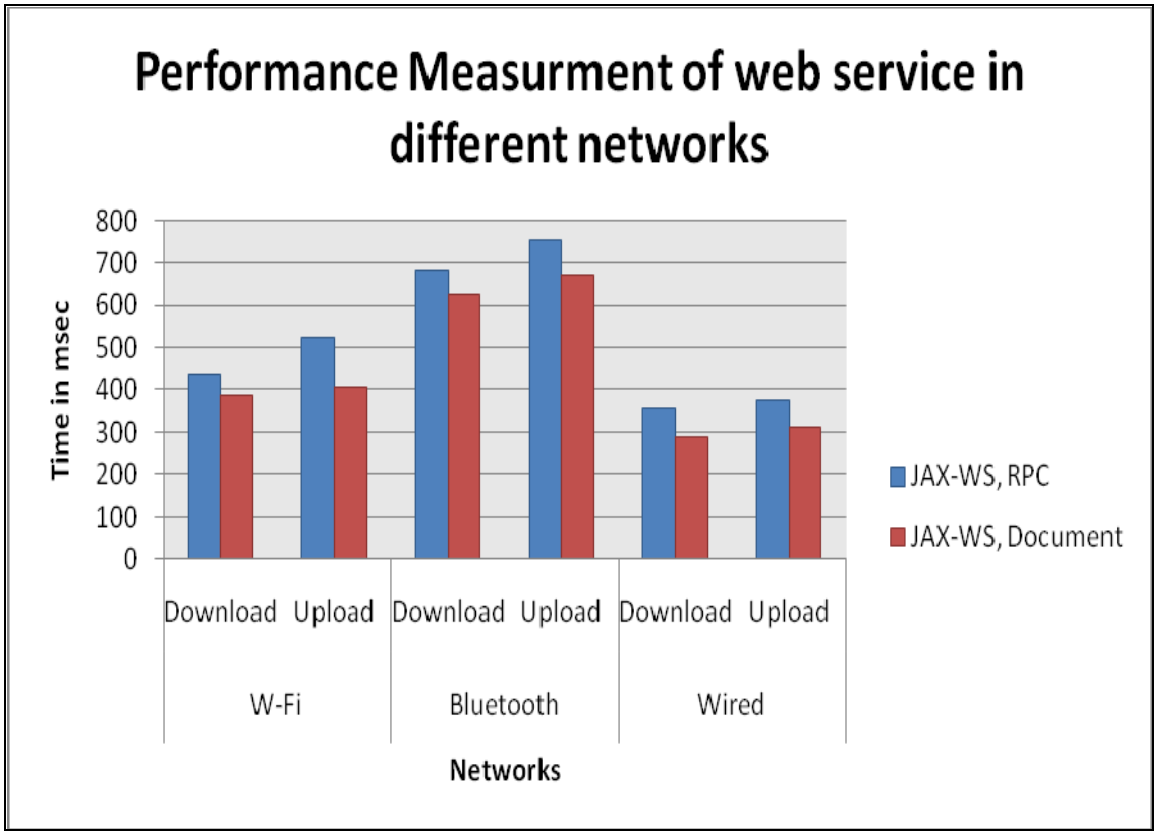

Figure 9. Performance measurement of Web service in different networks

V. CONCLUSIONS

We developed a web service which can transfer an image from client to server or vice-a-versa using MTOM technology. We implemented Web service using RPC style and Document style. It is observed that RPC style need almost 15% more time as that is required by Document style. The experiment is performed in three different networks, viz. Wi-Fi, Bluetooth and Wired and as expected web service work fast in wired environment. Same Web service can be developed using JAX-RS (REST) and performance of JAX-WS and JAX-RS need to be compared.

ACKNOWLEDGMENT

We wish to thank teachers of Department of Computer Science, Shivaji University, for their constant support and motivation. We also thank Principal of Thakur College of Science and Commerce, Mumbai, for allowing us to use resources of institute for performing the experiments.

REFERENCES

[1] A. Ng, P. Greenfield, and S. Chen. A Study of the Impact of Compression and Binary Encoding on SOAP Performance. Sixth Australasian Workshop on Software and System Architectures - Co-located with the Australian Software Engineering Conference - ASWEC'05, 01:46-56, 2005.

[2] H. Hass. Foundations and Future Directions of Web Services, 2005. Available at: http://www.w3.org/2005/Talks/1114-hh-ecows/. Last accessed on 3rd Jan 2013

[3] J. C. Estrella, F. J. Monaco, R. H. C. Santana, and M. J. Santana. Real Time Compression of Soap Messages in a Soa Environment. SIGDOC'08: Proceedings of the 26th Annual ACM International Conference on Design of Communication, pages 163-168, 2008.

[4] J. Evdemon. More thoughts on MTOM, 2005. Available at: http://blogs.msdn.com/jevdemon/archive/2005/05/07/415321.aspx. Last accessed on 3rd Jan 2013

[5] M. Ouzzani and A. Bouguettaya. Efficient Access to Web Services. IEEE Internet Computing, 8(2):34-44, 2004.

[6] Martin Gudgin, Noah Mendelsohn, Mark Nottingham, Hervé Ruellan, W3C Recommendation 25 January 2005, SOAP Message Transmission Optimization Mechanism http://www.w3.org/TR/soap12-mtom/, Last accessed on 17th Jan 2013

[7] S. Heinzl, M. Mathes, T. Friese, M. Smith, and B. Freisleben. Flex-SwA: Flexible Exchange of Binary Data Based on SOAP Messages with Attachments. In ICWS '06: Proceedings of the IEEE International Conference on Web Services, pages 3-10, Washington, DC, USA, 2006. IEEE Computer Society.

[8] SOAP Specifications, 2003 Available at: http://www.w3.org/TR/soap/ . Last accessed on 3rd Jan 2013.

[9] S. Heinzl, M. Mathes, T. Friese, M. Smith, and B. Freisleben. Flex-swa: Flexible exchange of binary data based on soap messages with attachments. In ICWS '06: Proceedings of the IEEE International Conference on Web Services (ICWS'06), pages 3–10, Washington, DC, USA, 2006. IEEE Computer Society

[10] W3C. Web Services Description Language (WSDL), 2001. Available at: http://www.w3.org/TR/wsdl. Last accessed on 3rd Jan 2013

[11] Y. Yang. Faster Data Transport Means Faster Web Services with MTOM/XOP, 2007. Available at: http://www.devx.com/xml/Article/34797, Last accessed on 17th Jan 2013.
[12] Y. Ying, Y. Huang, and D. W. Walker. A Performance Evaluation of Using SOAP with Attachments for e-Science. In UK e-Science All Hands Meeting (AHM), 2005.
[13] Zhang, P. Coddington, and A. Wendelborn. Binary Data Transfer Performance over High-Latency Networks Using Web Service Attachments. In E-SCIENCE '07: Proceedings of the Third IEEE International Conference on e-Science and Grid Computing, pages 261-269, Washington, DC, USA, 2007. IEEE Computer Society.