Invoking Web Service Using AJAX

G M Tere
Department of Computer Science,
Shivaji University, Kolhapur,
Maharashtra - 416004, ndia

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

B T Jadhav
Y.C. Institute of Science,
Satara, Maharashtra – 415001, India

ABSTRACT
This paper gives practical solutions for implementing Ajax, JavaScript, and Representational State Transfer (REST)–based Web services and functionality. We need to develop applications that are decoupled that is client code is separate from the server code. Using this approach applications can be easily tested and maintained. In some situations, responses to web service requests are not provided immediately, but rather sometime after the initial request transactions complete. Such asynchronous operations aren't explicitly supported by web services specifications and standards; however, those standards do include the infrastructure and mechanisms on which asynchronous operations can be based. Calling a web service asynchronously instead of performing a complete post back has several advantages, including less network traffic and generally elevated performance. Also with the support of DOM model and XMLHttp Object in major popular browsers, numerous client side AJAX libraries are available to make these calls. The paper examines and compares the typical and Ajax based invocation of web services with remedies to improve the performance of web services.

General Terms
Performance, Design

Keywords
AJAX, REST, request/reply operations, Web service, WSDL

1. INTRODUCTION
Asynchronous Services [8] are a fact of life, and a key requirement for successful SOA solutions. Many of the benefits of web services can't be realized until asynchronous interaction becomes well understood and widely practiced, and some high-value applications can't be deployed properly or at all without it. The simplest approach for implementing an asynchronous web service is to pursue a polling approach; the client makes a request, and then at a later time checks back to see if the response is ready. The advantage of this approach is that it is universally accessible. An AJAX [1,2] enabled web page could easily handle polling. Polling is probably the biggest waste of bandwidth we could devise. A more elegant approach for implementing an asynchronous web service is to pursue a callback approach; the client makes a request and leaves a return address (where the service should send the response). The obvious drawback to the callback approach is the limitation that it places on the client; the client must have a mechanism that allows it to receive messages.

1.1 What is a Web service?
The W3C defines a Web service as a software system designed to support interoperable machine-to-machine interaction over a network. This definition encompasses many different systems, but in common usage the term refers to those services that use SOAP formatted XML envelopes and have their interfaces described by WSDL. For example, WS-I only recognizes Web services in the context of these specifications. A Web service is a way of exposing application functions using the World Wide Web. It does this by using open protocols, so any consuming application that has access to the Web can also access the Web service. A typical Web service interaction consists of a consumer (the remote application that uses the Web service) sending an XML message to the service. The service then parses the message and sends back a response, also in XML format. The consumer then parses the response and, in some fashion, uses the information retrieved from the service.

Frequently, the XML language that is used in Web services is SOAP. SOAP was originally an acronym for Simple Object Access Protocol, but for some reason that is no longer the case. The protocol consists of three parts: the envelope, data type rules, and a means of defining operation requests and responses. Web services are defined with another XML document, known as a Web Services Description Language (WSDL). The WSDL specifies the operations that are exposed using the Web service, the data type definitions used by the operations, the protocol used to communicate with the Web service, and the location of the Web service itself. The advantage to Web services is that they enable applications that are written in different programming languages and deployed on different platforms to communicate with one another over the vast expanse of the Internet.

1.2 What is Ajax?
Ajax is one of the latest bleeding-edge technologies Web developers use to enable rich client presentation [13]. It accomplishes this by invoking a new request without disrupting the current view. An XML document is returned that is then displayed to the user, frequently as a subpage within the current presentation. In short, Ajax gives us the benefit of server-side dynamic content while still looking like client-side dynamic content. Ajax is a set of web and browser technologies that are used to create applications that look and feel more like desktop applications. Ajax technologies [9] are being used to build web applications with more responsive user interfaces, improved performance and higher levels of interactivity [4,6]. The core technologies behind Ajax are: dynamic HTML (DHTML), cascading stylesheets (CSS), JavaScript, and XML.

Ajax generally fulfills its billing through the use of the XMLHttpRequest DOM API, which, until the advent of Ajax, Web developers rarely used [9]. The request itself can be either one of the GET or POST varieties. As with any other request, a response is returned, which can be an error. If the response is not erroneous, the actual text of the response is used to update the current view.

AJAX provides for user the comprehensive and working well web application, enhances the user interaction experience [17]. In fact AJAX is not new and immeasurable technology, but combine many technologies existed: it uses JavaScript to operate DOM, to change and refresh user interface, then

continuously redraw and reorganize to show data for user, at the same time it dispose the user interaction base on mouse and keyboard. CSS provide for user a uniform appearance and make a powerful shortcut for programming to operate DOM. By XMLHttpRequest object to communication asynchronously with the web server and submit user's request and get new data. With asynchronous services, clients initiate a request to a service and then resume their processing without waiting for a response. The service handles the client request and returns a response at some later point, at which time the client retrieves the response and proceeds with its processing. The traditional web application is based on full page refreshing, but AJAX application only transfers and updates the data needed, not full page. This work model increases the response speed of user request, and avoids screen flicker causing by page refreshing, eventually brings good user experience as desktop applications.

With asynchronous services, the client invokes the service but does not wait for the response. Often the client does not want to wait for the response because it may take a significant amount of time for the service to process the request. The client can continue with some other processing rather than wait for the response. Later, when it does receive the response, it resumes whatever processing initiated the service request. The model of traditional web application and AJAX are shown in Fig 1.

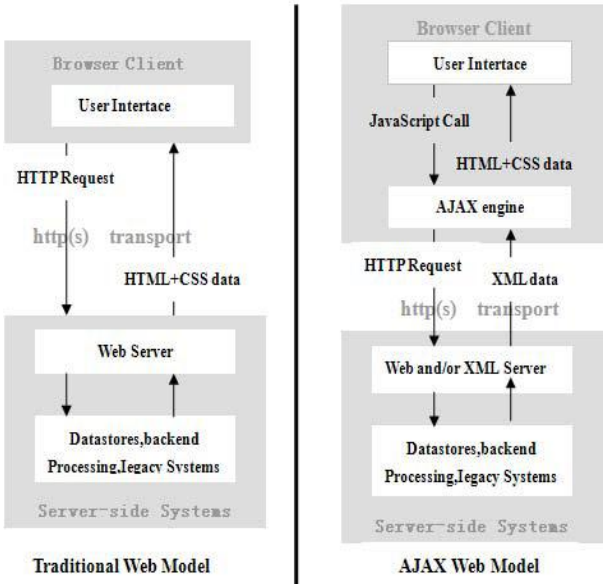

Fig 1: Traditional and AJAX Web model

2. AJAX and WEB SERVICES

Think of it this way: a rich client experience coupled with services accessible anywhere on the Internet. Ajax executes a request under the covers and usually spits out the response back to the Web page without an entire refresh of the page. Now, while that request can be a simple HTTP [14] request, it can also be a SOAP message sent to an exposed Web service. The JavaScript side of the Ajax routine can then parse the response (also in SOAP format) and extract the necessary data that is returned to the application and presented to the user.

Ajax and web services are a perfect match for developing web applications. Ajax has built-in abilities [9,11] to access and manipulate XML data, the native format for almost all REST and SOAP web services. When developing an Ajax and REST [3] application, one must decide on the tools and frameworks to be use. The choice is simple: Use whatever you are using today, and write some Ajax applications. There is no need to change the tools being used. We can use tools like ASP.NET, JavaServer Pages (JSP), PHP, Ruby, or Python. Ajax uses JavaScript, DHTML, and the XMLHttpRequest object. Fig. 2 shows how to decouple the client from the server.

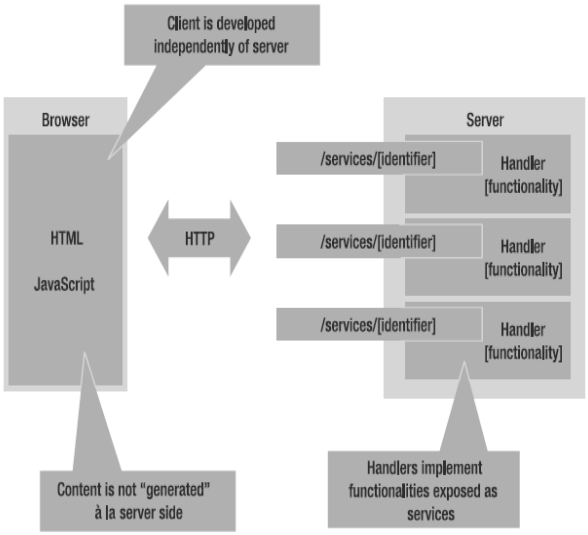

Fig 2: Decoupling client from the server

Having decoupled the client from the server, one can make use of specific frameworks to make it simpler to implement specific pieces of logic. For example, a client-side developer could use the Prototype or Dojo frameworks.

When getting started with Ajax and REST [3], remember the following points:

We can use Ajax and REST today with existing technologies. We generally don't need to throw out old technologies and replace them with new ones.

Ajax and REST are about decoupling the client from the server and making using of Web services.

Ajax and REST frameworks [12]can make it simpler for us to implement our applications, but because there are so many frameworks, we need to inspect them to see if they fulfill our needs.

Web Services have given Enterprise computing a new dimension for performing business activities. They provide interoperability between numerous business processes and also brings automation of these tasks for various organizations. Typically web services were designed and created to provide data to Rich Internet Applications. They can be invoked or called by a number of web sites. Professional web services like Yahoo weather, Stock indexes etc are the best suited examples for the implementation of Web Services.

Typically web services were planned to call with the synchronous behavior i.e. the client/user has to wait for the response from the server and cannot continue with his/her work. But with the growth of web based applications over a couple of years and implementation of standards like Web 2.0, Asynchronous invocation [8] of the web services in distributed computing platform has seen a tremendous rise and acceptability in the web based programming. By asynchronous service model, we mean the client can continue working on other tasks without waiting for the result from the

server after it calls a service; the service will inform the client about the results of the call at a later point in time.

Although XML is the open and widest structural data transmission method, for the web developer there are other technologies to select. AJAX is one of these. The kernel of AJAX is XMLHttpRequest, which made it easily to access web services. We can use JavaScript Object Notation (JOSN) to encapsulate data and decrease data traffic. In fact we also can use custom format to invoke web services by XMLHttpRequest, and parse SOAP message to get data.

2.1 Software Requirements

All the examples in this paper are developed using the RC build of ASP.NET AJAX that one can download from ajax.asp.net. Moreover, we need to have SQL Server 2005 with the Northwind database installed. The examples use Visual Studio 2005 as the IDE.

2.2 Example Scenario

The example develops a web form that acts as a data entry page for an Employees table of the Northwind database. Using ASP.NET AJAX features, this data entry page will consume a web service that allows us to SELECT, INSERT, UPDATE, and DELETE employees.

Now that we have created a new web site, add a new web service to it and name it EmployeeService.asmx. The EmployeeService will contain five web methods as shown in Table 1.

Table 1. Web Methods in EmployeeService

Method Name	Description
GetEmployees()	Returns a list of employees from the Employees table. The list is returned in the form of an array of Employee objects.
GetEmployee()	Accepts an EmployeeID and returns its details as an Employee object.
Insert()	Adds a new employee to the Employees table.
Update()	Updates an existing employee from the Employees table.
Delete()	Deletes an existing employee from the Employees table.

The GetEmployees() and GetEmployee() methods return data in the form of Employee object(s). Therefore, we first need to create a class called Employee.

3. IMPLEMENTATIONS

We have implemented the project with following steps:

a) Creating the Asynchronous Web Service

```
using System.Linq;
using System.Web;
using System.Web.Services;
using System.Web.Services.Protocols;
using System.Xml.Linq;
[WebService(Namespace = "http://tempuri.org/")]
[WebServiceBinding(ConformsTo =
WsiProfiles.BasicProfile1_1)]
// To allow this Web Service to be called from script,
using ASP.NET AJAX, uncomment the following
line.
[System.Web.Script.Services.ScriptService]
public class Service : System.Web.Services.WebService
{
public Service () {
}
[WebMethod]
[ScriptMethod()] >_
public DataSet GetEmployees(String vCity)
{
//code to get all the employees from the database
parameter of City
}
}
```

b) Setting the Configuration to use Proxy in Javascript

```
<asp:ScriptManager ID="ScriptManager1"
runat="server">
<Services>
<asp:ServiceReference path="~/Service.asmx" />
</Services>
</asp:ScriptManager>
```

The above code helps in setting a proxy registration of the web service class in the web form, so that it can be invoked from JavaScript asynchronously. Numerous frameworks re available to do the same with little bit of difference in the coding style.

c) Writing Delegate Code in JavaScript to handle Asynchronous calls

The third step includes the writing of delegate that will handle Callback. This is primarily done at the JavaScript level that makes the client calls without complete request-response cycle.

```
<script language="javascript">
function CallWebService()
{
Service.GetEmployes("XYZ",OnCallBack);
}
function OnCallBack(result)
{
if(null!=result)
{
$get("DataGrid").DataSource=result;
$get("DataGrid").DataBind();
}
}
</script>
```

The code contains two methods i.e. CallWebService which is called on the clicking of the button and also intialises a CallBack delegate known as "OnCallBack". This delegate is defined as a separate function in JavaScript by the name of OnCallBack(result) and it takes one argument called 'result'. This receives the data from the web service method that has been invoked. The web service is invoked with the client end function and data is returned in the form of DataSet that can be manipulated in any form. The user does not have to wait for the form to post back i.e. complete request-response cycle rather the user continues with the work while the partial update is transferred from the web server to the client. The above implementation explains the web services and their

invocation with help of asynchronous calls. This helps in the following:

Elevated performance of the web server because the complete web form posting is not done and only partial updates are retrieved from the web server.

Results in higher bandwidth and optimal utilization of the same which enhances the user experience.

4. Testing the Web Page

Now that we have developed the web service and the client application, it's time to test them. Run the web form and try inserting, updating, or deleting employees. Fig 3 shows the web form after updating an employee.

4.1 Calling External Web Services

In this example, the EmployeeService was part of our web site. Sometimes, we may need to call web services that are not hosted by our domain at all. ASP.NET AJAX internally relies on the XML HTTP object, which for security reasons cannot communicate outside the originating web site. That means the above technique won't work for external web services. Unfortunately, ASP.NET AJAX does not have a direct solution to this issue. However, Microsoft has released a "bridge" technology. We can use the bridge to call a wrapper class residing in our web site, which in turn calls the actual web service. In the current RC version, we can create a wrapper web service in our web site, which in turn calls the original web service. From the client application, we then can call our wrapper web service and achieve the communication. The following is an outline of the necessary steps:

Fig 3: Web Form After Updating an Employee

a) Add a web reference to the external web service in our web site.
b) Create a new web service in our web site.
c) In the newly created web service, provide wrapper web methods that call the external web methods via 0web reference.
d) Call this newly added web service from the client application as described in this paper.

4.2 Infrastructure for Calling ASP.NET Web Services

ASP.NET AJAX provides a complete infrastructure for calling ASP.NET web services from client-side JavaScript. We can easily integrate server data with the user-responsive web pages using AJAX. The ASP.NET AJAX framework generates a JavaScript proxy for our web service. The proxy then is used to call the web methods.

5. CONCLUSIONS

Web services is especially suitable for data communications over Internet and it is a best solutions for enterprise integrating heterogeneous system and data-sharing. However it is a focus item to improve the performance of web services effectively. Simple mechanisms that support asynchronous communication patterns can be constructed using current technologies available for Web services development. These mechanisms are nearly identical to those already used for Web applications, as demonstrated in the simple examples presented in this paper. Web services and a service-oriented style of architecture are widely seen as the basis for a new generation of distributed applications and system management tools. This paper introduces the key concepts, benefits of asynchronous calls and methods to invoke web services to improve the overall user experience and higher performance. Paper gives the implementation of the asynchronous based frameworks and implementation example in ASP.NET to justify the concept. With the rising popularity of Web 2.0 and Rich Internet Based Applications, callbacks/asynchronous calls are making the implementation of the web services more powerful and help in improving the functioning of the web server. By using some of the techniques presented in this paper, we can make our Web service applications implement logic needed for an asynchronous communication model, and ease our future migration pathway to standards-based asynchronous messaging constructs when they become available.

6. ACKNOWLEDGMENTS

We thanks teachers of Department of Computer Science, Shivaji University, Kolhapur for motivating us for this research work. We wish to thank the Principal of Thakur College of Science and Commerce, Mumbai and Principal, Y.C. Institute of Science, Satara for providing resources required to complete this paper.

7. REFERENCES

[1] Anthony T. Holdener III, Ajax: The Definitive Guide, O'Reilly Media, 2008

[2] Brett McLaughlin, Mastering Ajax, Part 2: Make asynchronous requests with JavaScript and Ajax, IBM developerWorks, Jan 2006

[3] Christian Gross, Ajax and REST Recipes: A Problem-Solution Approach, Apress, 2006

[4] Dyachuk, D.; Deters, R.; , "Optimizing Performance of Web Service Providers," Advanced Information Networking and Applications, 2007. AINA '07. 21st International Conference on , vol., no., pp.46-53, 21-23 May 2007

[5] Erik Christensen, Francisco Curbera, Greg Meredith, Sanjiva Weerawarana, Web Services Description Language (WSDL) 1.1, http://www.w3.org/TR/wsdl

[6] Fangju Wang; , "A High Performance Architecture for Web Service Systems in XML," Service Systems

and Service Management, 2007 International Conference on , vol., no., pp.1-6, 9-11 June 2007

[7] G. Alonso, F. Casati, H. Kuno, V. Machiraju, Web Services Concepts, Architectures and Applications. Springer Verlag, October 2003

[8] G. M. Tere, B. T. Jadhav, R. R. Mudholkar, "Web Services using Asynchronous Communication", Proceedings of National Conference on Advancement of Technologies - Information Systems & Computer Networks, ISCON 2012, 3-4 Mar 2012, GLA University, Mathura and CSI and IETE.

[9] Jianbo Bai; Hong Xiao; Tianyu Zhu; Wei Liu; Aizhou Sun; , "Design of a Web-Based Building Management System Using Ajax and Web Services," Business and Information Management, 2008. ISBIM '08. International Seminar on , vol.2, no., pp.63-66, 19-19 Dec. 2008

[10] Kasse, Y.; Mokdad, L.; Sene, M.; , "Performance analysis of Web services architecture," Computers and Communications, 2009. ISCC 2009. IEEE Symposium on , vol., no., pp.94-98, 5-8 July 2009

[11] M. Brambilla, S. Ceri, S. Comai, P. Fraternali, Model driven development of Web Services and hypertext applications. SCII03, June 2003, Orlando

[12] M. Ruth, F. Lin and S. Tu. "A Client-side Framework Enabling Callbacks from Web Services", In Proceedings of the European Conference on Web Services (ECOWS), pp. 105-116, 2005.

[13] Mark Pruett, Ajax and Web Services, O'Reilly Media, August 2006

[14] Matt Powell, Asynchronous Web Service Calls over HTTP with the .NET Framework, Microsoft Corporation, 2009

[15] Vieira, M.; Laranjeiro, N.; , "Comparing Web Services Performance and Recovery in the Presence of Faults," Web Services, 2007. ICWS 2007. IEEE International Conference on , vol., no., pp.623-630, 9-13 July 2007

[16] Xiao-yang Guo; Xue-song Qiu; Ying-hui Chen; Fan Tang; , "Design and implementation of performance testing model for Web Services," Informatics in Control, Automation and Robotics (CAR), 2010 2nd International Asia Conference on , vol.1, no., pp.353-356, 6-7 March 2010

[17] ZhenXing Chen; , "Using AJAX to Optimize Web Services Performance," Computational Intelligence and Software Engineering, 2009. CiSE 2009. International Conference on , vol., no., pp.1-4, 11-13 Dec. 2009