AJAX Integration Approach for Collaborative Calendar-Server Web Services

Ahmet Fatih Mustacoglu[1, 2], and Geoffrey Fox[1, 2]
[1]Community Grids Lab, Indiana University, Bloomington, IN, 47404, USA
[2]Department of Computer Science, Indiana University
{amustaco, gcf}@cs.indiana.edu

ABSTRACT

The proliferation of the impressive web technologies provides developers with a way to produce performance efficient, usable, rich and interactive web based applications. A set of new technologies called AJAX (Asynchronous JavaScript and XML) for the browser based applications is gaining the web developer's interest. In this paper we present a generic and performance efficient framework for integrating AJAX models into the iCalendar based Collaborative Calendar-Server Web Services system.

1. INTRODUCTION

AJAX [1] is getting more popular for the web based application developments. It is not a single technology that does the magic for applications. It is a combination of technologies for the web based applications. Some of the standard technologies that composed the AJAX model are JavaScript, XML, CSS, DOM and XSLT. Gmail [2] and Google Maps [3] for example are the high performance AJAX applications developed by Google.

Web Services [4] increase the level of interoperability between different applications running on different platforms. Web Services are self-descriptive, self-contained and modular systems, and they are defined by the Web Services Description Language (WSDL) [5]. Web Service systems support XML based message exchange mechanism, which enable us to be able to develop loose coupled distributed systems all around the world. They use Simple Object Access Protocol (SOAP) [6] for message exchanging. We believe that the AJAX model and the Web Service Systems could support and improve each others strength since they are both using XML based messaging structures.

In this paper, we first give some background information regarding to the technologies that we have been using in our proposed frame work. These technologies consist of the AJAX model, the Web Services, Grid Services, and iCalendar [7] based Collaborative Calendar-Server (CCS) Web Services [8]. In Section 3 we mention about our design and architecture for our proposed framework. Next, we are going to give a comprehensive explanation of our implementation details in Section 4. Then, we are going to present our performance test results using the AJAX model and traditional model in Section 5. Finally, we are going to give the future work and conclusion in Section 6.

2. BACKGROUND

Rich applications are getting more popular due to their impressive features including high performance (compared to earlier systems), responsiveness, and interactive capabilities. Traditional web applications are generally suffer from slow performance and limited interactivity compared to the AJAX based applications. AJAX based applications are shows a better performance since it can add or retrieve a requested data without reloading the page. For example, AJAX-based Google Maps beta updates the page almost instantly when it receives the request from users. With, a standard web application, the page need to be reloaded to update the page and meanwhile users need to stare at a blank page [9]. In AJAX model, asynchronous data retrieval is done through the XMLHttpRequest, interactions and dynamic display nature is succeed by using the Document Object Model (DOM). XHTML and CSS provide a mechanism for standards-based presentation of the system, and data modification and exchange is done by using XML and XSLT. Finally, JavaScript binds everything together in an AJAX based application system.

After Google Maps, Yahoo [10] has also released its AJAX-based Map. This two giant's impressive developments with Google Maps and Yahoo Map are triggering web based application developers' attention.

AJAX-based systems can be implemented in client applications by writing JavaScript codes that directly use the XMLHttpRequest protocol's API to communicate with the server or the Web Service. However, there is some compatibility issues with the browser, so these

issues should be keep in mind during the design and development period.

Our proposed AJAX-based framework is a calendaring and scheduling services implementation model that provides users with the ability of using services provided by the iCalendar based CCS Web Services System [8]. Web Services can be accessed through the message exchange mechanism in SOAP format. By combining Web Services, which use SOAP over HTTP, with the XML based structure AJAX-model in our framework, we are planning to improve the performance and to leverage the level of interoperability for different kind of applications running on a different platform for our proposed system.

3. DESIGN AND ARCHITECTURE OVERVIEW

Our proposed system has been designed to interact with CCS Web Services from an AJAX-based interface to provide calendaring and scheduling services over the internet. In our framework, users interact with the Web Services through the pure JavaScript codes. User interface is implemented by using HTML and JavaScript. The CCS's Web Services can be executed from our JavaScript file called "calendarservice.js", and we have used "ws.js" from IBM Corporation [11] and "prototype.js" framework by Sam Stephenson [12] for generating WS.Call object. In our framework, users first need to be authenticated with the system in order to post or retrieve calendar information from the CCS system. Having successfully authenticated with the CCS system, users can access and post an event into their private calendar, which can only be seen by its owner. They can also schedule a meeting into the public calendar, which can be accessed by the entire registered user with the CCS system. All the access to the calendars has been synchronized. An overview of our framework architecture design is depicted in Figure 1.

Users' login information is stored in MySQL Database [13], and this information is used for authenticating the users with the CCS system. Basically, our JavaScript called "calendarservice.js" code executes the coming request from users by using "ws.js" and "prototype.js" JavaScript files to setup the WS.Call object to call the associated Web Service of the CCS. Each service of the CCS system first tried to authenticate the user login information. If users' login information is correct, then the requested service is provided and the result is returned to the client. Otherwise, called Web Service of the CCS returns a warning message to the users to let them know that they need to register in order to gain access into the CCS Web Services.

With the AJAX-based friendly user interface, users' requests and Web Service responses can be displayed by just updating the requested information within the browser almost instantly without reloading the entire page, which basically reduces the performance of the whole system.

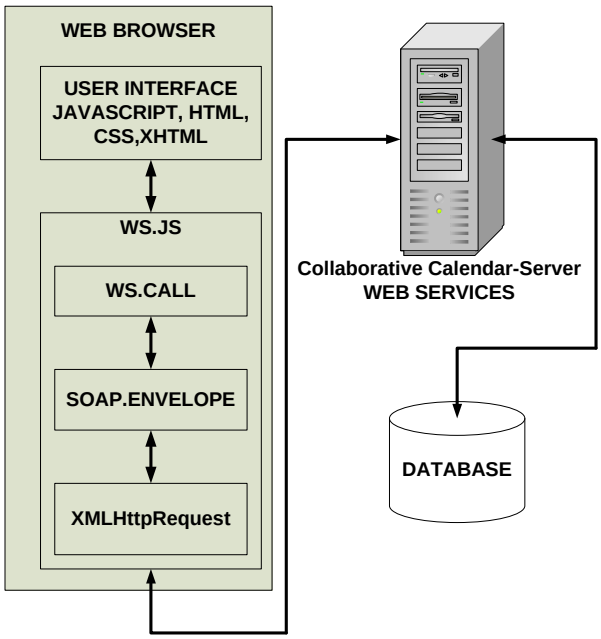

Figure1. Invoking the CCS Web Services from the AJAX Application

4. IMPLEMENTATION DETAILS

Our AJAX-based interface [14], which interacts with the CCS Web Services, has been implemented by using the HTML, and JavaScript technologies. CCS Web Services are being called from the pure JavaScript code based on the users' request, and results displayed within the browser almost instantly without reloading the entire page.

In our calendarservice.js file, we have currently implemented "doregister, publicCalendar, privateCalendar, addEvent, scheduleMeeting" methods to communicate with the CCS Web Services. These methods generate the WS.Call object and WS.Call invokes the CSS Web Services in RPC style. Sample code for generating WS.Call object to invoke the CCS Web Service is as follow:

- var call = new WS.Call('http://gf8.ucs.indiana.edu:18086/AJAXWSCalendar/services/IcalCalendarServer?wsdl'

- call.invoke_rpc(qn_op, new Array({name:'username',value:username},{name:'password',value:password},{name:'name',value:name},{name:'lastname',value:lastname},{name:'email',value:email}),null,function(call,envelope){var ret = envelope.get_body().get_all_children()[0].get_all_children()[0].get_value();});

doregister: This functionality allows users to register with the CCS framework. When users click on this functionality, a popup window is opened. In this popup window, users need to enter the username, password, name, last name, and email information in order to register with the system. Once users click on the "Click to register" button, doregister method is being called. After generating WS.Call object in doregister operation, the users' registration process is completed by inserting those fields into the Database.

publicCalendar: This method, shows the public/collaborative calendar within the browser. It receives username, password, and container variables, and executes the associated Web Services of the CCS to retrieve the public calendar. If the login procedure with the system is successful, this function sets the associated container with the requested calendar data within the browser to show the public calendar information to the user. Otherwise, returns a warning to the users so that they can register with the system first in order to gain access to the system.

privateCalendar: Users can access their private calendar via this option of the AJAX-based interface. This menu item receives username, password, container parameters, and brings the user's private calendar via the necessary service calls, if the login information is correct. If users login is not authenticated by the CCS Web Services, then users asked to register before accessing the system.

addEvent: Via this method, users can add/insert new events into their private calendar. This method receives username, password, event start date, start hour, start minute, start time am/pm value, event end date, end hour, end minute, end time am/pm value, public or private value for this event, first name, last name, email, event name, event location, and event description information from the users. Users can complete the operation by pressing on the "Add Event" button.

scheduleMeeting: Users can schedule a meeting into the collaborative/public calendar via this option, and results displayed within the browser. This method requires, username, password, event start date, start hour, start minute,start time am/pm value, event end date, end hour, end minute, end time am/pm value, first name, last name, email, event name, event location, and event description information. Once, the users click on the "Schedule a Meeting" button, this JavaScript method from our calendarservice.js is called, and it executes the related Web Service based on the request.

5. TEST RESULTS

We have performed several tests to evaluate our proposed framework, and we have calculated turnaround time, standard deviation, and standard error values for our proposed AJAX model and traditional request/response systems. The comparison of two systems and environment settings are given as below. And calendar file size user for these tests is 32KB.

In each time, our web service has been called 100 times to measure the turn around time, and it has been called 1000 times as a total. Summary of testing environments is depicted in Table 1.

Table 1. Summary of Environment and Machine Configurations

Axis: Running on GridFarm8	
Processor	Intel® Xeon™ CPU (2.40 GHZ)
RAM	2GB total
Bandwidth	100Mbps
OS	GNU/Linux (kernel release 2.4.22)
Java Version	Java 2 platform, Standard Edition(1.5.0_01)
SOAP Engine	AXIS 1.2 and Tomcat 5.0.28

AJAX-based Service Client	
Processor	Intel Pentium 4 CPU 3.40GHz
RAM	1.00 GB total
Bandwidth	100Mbps
OS	Windows XP Professional
Browser	Internet Explorer 7.0

Traditional-based Service Client	
Processor	Intel Pentium 4 CPU 3.40GHz
RAM	1.00 GB total
Bandwidth	100Mbps
OS	Windows XP Professional
Browser	Internet Explorer 7.0

In Figure 2, we have calculated the average turnaround time for our proposed AJAX-based framework for retrieving the users' private calendar, which is 32KB in size. In each test case, our service returns the requests most of the time in 0 millisecond time. We have also

measured the Standard Deviation and Standard Error values for our proposed AJAX-based framework. Standard Deviation test results are given in Figure 3, and Standard Error test results are given in Figure 4.

In Figure 5, we have calculated the average turnaround time for the traditional framework, which works request and response based, for retrieving the users' private calendar, which is 32KB in size. In each test case, the CCS Web Service returns the requests average in 367 milliseconds. We have also measured the Standard Deviation and Standard Error values for the traditional framework. Standard Deviation test results are given in Figure 6, and Standard Error test results are given in Figure 7.

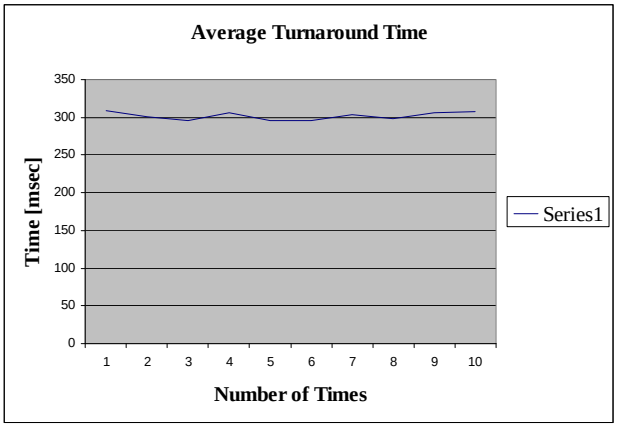

Figure2. Average Turnaround Time for AJAX Model

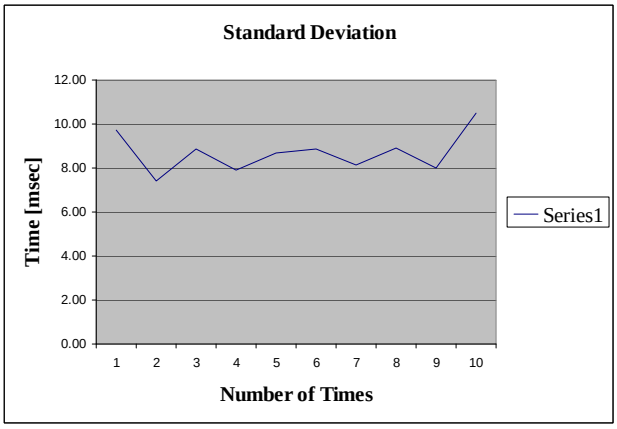

Figure3. Standard Deviation for AJAX Model

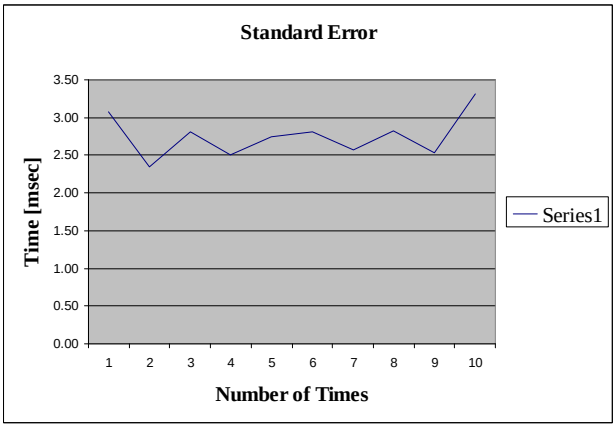

Figure4. Standard Error for AJAX Model

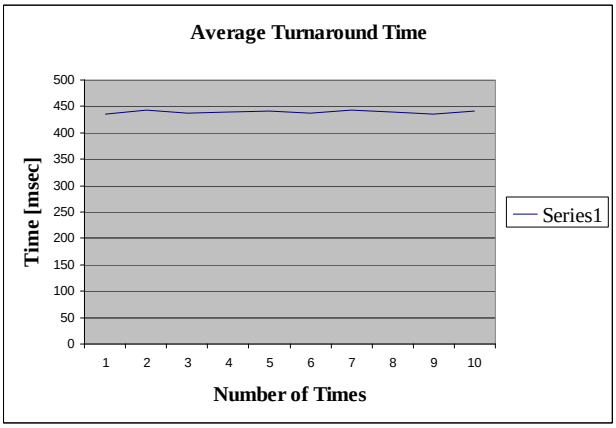

Figure5. Average Turnaround Time for Traditional Model

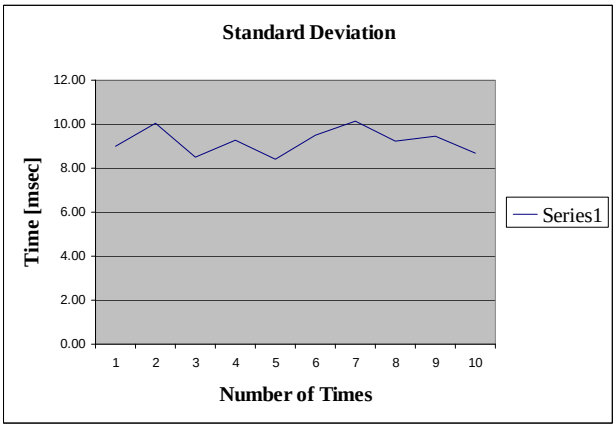

Figure6. Standard Deviation for Traditional Model

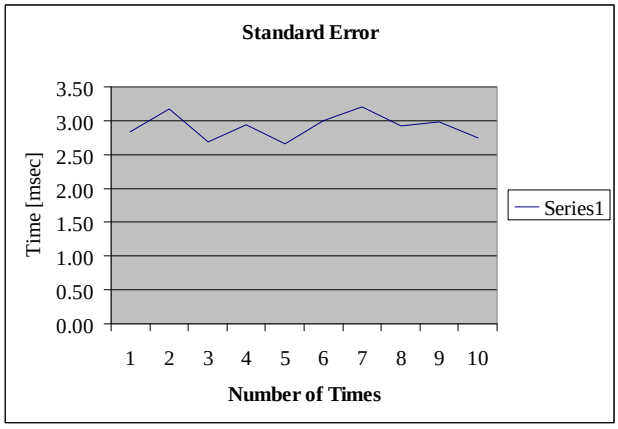

Figure7. Standard Error for Traditional Model

When we compare the turnaround time as a performance issue, our proposed AJAX-based frame work is much faster than the traditional model for retrieving users' private calendar from the CCS.

6. CONCLUSION AND FUTURE WORK

We have developed a calendar server model called Collaborative Calendar-Server Web Services [8] to provide calendaring and scheduling services over the internet. CCS has been implemented as a pure Web Services by using Java Language and Java Servlet Technology. We have integrated a calendaring and scheduling client into the GlobalMMCS Portal [16] developed by Community Grids Laboratory at Indiana University to access the CCS's Web Services in GlobalMMCS portal for calendaring and scheduling needs.

Web Service Technology provides a new level of interoperability between different platforms and languages. By taking advantage of Web Service Technology, we have developed and implemented a calendaring and scheduling client, which can communicate with CCS's Web Services via AJAX Technology. In our proposed framework here, we have proved that our Collaborative Calendar-Server Web Services can also be accessed via AJAX based models by simply calling the associated CCS Web Service's interfaces in JavaScript by using calendarservice.js" written by us, "ws.js" from IBM Corporation [11] and "prototype.js". We have also measured turnaround time, standard deviation, and standard error values for our proposed AJAX model and traditional request/response systems. As expected, AJAX based model brings the result faster than the traditional request/response model.

As a result, our proposed performance efficient AJAX based framework can be integrated into the CCS Web Services [8] system, and the CCS Web Service [8] system is provides a framework for implementing calendaring and scheduling applications to access its services from different languages and platforms.

RERERENCES

[1 Tom Noda, Shawn Helwig, Rich Internet Applications http://www.uwebc.org/docs/final_1.pdf

[2] GMail Website http://www.gmail.com

[3]Google Maps Beta Website http://maps.google.com/

[4] Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C., and Orchard, D. "Web Service Architecture." W3C Working Group Note, 11 February 2004. Available from http://www.w3c.org/TR/ws-arch

[5] Erik Christiensen, Francisco Curbera, Greg Meredith, Sanjiva Weerawarana, Web Service Description Language (WSDL) Version 1.1, March 2001. Available at http://www.w3.org/TR/wsdl

[6] Don Box, David Ehnebuske, Gobal Kakivaya, Andrew Layman, Dave Winer, Simple Object Access Protocol (SOAP) Version 1.1, May 2000. Available at http://www.w3.org/TR/2000/NOTE-SOAP-20000508

[7] Internet Calendaring and Scheduling Core Object Specification Web Site: http://www.ietf.org/rfc/rfc2445.txt

[8] Ahmet Fatih Mustacoglu, Wenjun Wu, and Geoffrey Fox Internet Calendaring and Scheduling Core Object Specification (iCalendar) Compatible Collaborative Calendar-Server (CCS) Web Services IEEE 2006 International Symposium on Collaborative Technologies and Systems CTS 2006 conference Las Vegas May 14-17 2006

[9] Jesse James Garret, Ajax: A New Approach to Web Applications http://www.adaptivepath.com/publications/essays/archives/000385.php

[10] Yahoo Website http://www.yahoo.com

[11] IBM Corporation http://www.ibm.com

[12] Prototype JavaScript framework http://prototype.conio.net

[13]MySQL Website http://www.mysql.com/

[14] Proposed framework interface http://gf8.ucs.indiana.edu:18086/AJAXWSCalendar/index.jsp

[15] Collaborative Calendar-Server Project web site
http://www.opengrids.org/wscalendar/

[16] Global Multimedia Collaboration System Web Site:
http://www.globalmmcs.org/