Web-Enabled Remote Laboratories

K. Ganesan, B.S. Saranraj
TIFAC-CORE in Automotive Infotronics,
VIT University, Vellore 632014
M. Tech in Computer Science and Engineering,
VIT University, Vellore 632014
INDIA
kganesan@vit.ac.in

Abstract: This paper describes about accessing the real laboratories using web services. Students doing the course on "Embedded systems" can perform the various experiments such as analyzing the frequency measurement, generation of pulse width modulation using Freescale-semiconductor’s kit and Digital Storage Oscilloscope (DSO) by giving different input parameters from anywhere at anytime through Internet or Intranet using a webpage provided in a client browser and submit it to a lab server (LS), which is interfaced with Freescale-semiconductor’s kit and digital storage oscilloscope (DSO). The experimental specification is forwarded to the lab server through a service broker. Based on the experimental specifications given by the user, experiments can be conducted on the lab server side using real equipment interfaced with the LS. The application can be executed and the output generated on the DSO can be captured by the lab server machine and one can even convert the results into XML file and send those results to service broker. If user needs, the results can even be displayed on the remote client machine by the service broker. The present work uses iLab architecture which provides three tier client / broker / server architecture for the development and deployment of remote labs.

Key-Words: Lab Server, Remote Labs, Service broker, Web services.

1. Introduction

Remote laboratories are real equipment laboratories that can be operated and controlled remotely through an interface. Most of the institutions and research laboratories do not own all the costly equipment to provide lab facility for all the users to conduct their experiments or to use devices. In some institutions, students are lacking practical experience due to lack of equipments. To solve these problems, there are different techniques available such as simulation of experiments, virtual instrumentation, etc. Simulation of experiments does all the steps or procedures using software packages but no hardware device is involved in this simulation process. The virtual instrumentation technique does the same using hardware modules. If one wants to conduct an experiment using a real device, there are many approaches to design an Internet or Intranet based remote labs. Early systems required specialized platform dependent software to be running on client machine to provide remote lab facility. Sometimes later browser enabled software including dynamic HTML, CGI, Java applet, etc., were used at the client side based on specific experimental requirements. If the client is using HTML, only little processing capability can be provided at client end and bulk of the processing has to be done at the server side system. Real lab concept can provide facility to pass only input parameters required to conduct a complete experiment from anywhere through Internet / Intranet.

The iLab architecture provides a platform independent framework for remote lab development and deployment. Usually, the nature of each lab experiment is disparate, and discipline-specific. Initially researchers have developed online experiments and separated the details of student, data, and resource management. But each experiment developed has its own approach to validate the students, scheduling their sessions with the online lab, and storing or returning the experiment results. Teams working on
separate experiments frequently duplicated each other’s efforts. In order to avoid the unstable and insecure software mechanisms, the same software architecture is to be followed in all the experiments during development process.

One can perform experiments on devices from remote machines by providing the required parameters or input specifications, to conduct the experiment, called experimental specification. During this process, one has to use some APIs to offer secure administration, authentication, authorization, and data storage. The list of APIs used includes Service Broker Administrative API [1], Service Broker Authentication API [2], Service Broker Authorization API [3], Service Broker Data storage API, Client to Service Broker API [4], and Service Broker to Lab server API [5].

![Proposed Architecture of Web-Enabled Remote Laboratories](image)

All the modules of the present work use the same web services framework and Simple Object Access Protocol (SOAP). The use of SOAP and web services framework allows the client and lab server to use any platform. Service Broker will take care of all the administration, authorization, authentication, data storage at a central location in order to conduct the experiment. Client side machine need not have the same platform and software as lab server. Real equipment or costly device is connected with the lab server through appropriate interfacing. Device or equipment damage can be avoided by validating the experimental parameter specifications before sending it to the lab server. The entire system has been developed using Web Services framework to enable anywhere access. After submitting the experiment specification, the user need not even stay on line. If the experiment is successfully completed, at anytime the user may be able to see the results by placing a request to SB. If the user wants, experimental execution can be captured via live streaming video and the same can be displayed on the user screen. But in order to utilize this option, the user needs to stay online.

2. Related Works

Some of the APIs and general architecture developed by Microsoft Research Labs and Massachusetts Institute of Technology (MIT) has been used to perform administration, authentication, authorization, and data storage at central location. There are only few experiments that are made accessible from anywhere using this iLab architecture of MIT. The experiment, called Microelectronics Web Lab [6], provides an online device
characterization experience for microelectronics students. We use Java-based lab client [7] due to various graphics factors involved in the experiments. It allows the student to use dynamic signal analyzer for the measurement of electronic, electromechanical, mechanical, and thermal systems.

**Figure 2. Design topology of A Web-Enabled Remote Laboratories as implemented within iLab shared architecture**

### 3. Software Architecture

**Overview**

There are two different types of architectures available in iLab such as Batched Architecture and Interactive Architecture. Batched architecture needs all the specification for the experiment at one stretch before starting the experiment execution process and user need not stay online until the completion of execution. Interactive architecture requires input specification during runtime. A Web-Enabled Remote Laboratories integrates iLabs redesigned batched experiment architecture to collect all the input parameters before starting the experiment and optional streaming feature, and to access this facility the user needs to stay online. In this paper, we use independent three tiers namely Lab Client, Service Broker and Lab Server. Communications between these three tiers is via web services with SOAP as a communication protocol as shown in Fig.1.

The use of web services and SOAP allows the client to use any platform and can be implemented in any programming language. The only requirement for each tier is that it should conform to iLabs batched architecture APIs. The first tier, called lab client is implemented in ASP.NET [8]. It uses a windows based application that communicates with the service broker. The second tier i.e., service broker, communicates with the lab server through web services, using SOAP as a communication protocol. The Lab server uses web services application for interfacing with relevant Freesacle Semiconductor kit and Digital Storage Oscilloscope for frequency generation experiment and for pulse width modulation experiments.

**Figure 3. Component diagram of lab client interface – Web-Enabled Remote Laboratories**

In figure 2, the design methodology of Accessing Real Labs using Web Service (ARL) [9] is shown. All communications between the client and lab server should be mediated by the Service Broker (SB). If communications are routed through SB, this would save an authoritative log of the user’s control of the experiment and the corresponding results. In the lab side at
the campus, the Freescale semiconductor kit can be connected to the lab server using appropriate hardware interface. The client is going to control the experiment from the client side using his system. Freescale’s Microcontroller kit is interfaced with lab server. Through that kit many experiments, for courses on embedded system, can be accessed by students. Client gives the input parameters for the experiment at client node. SB will have all the administrative services such as user authentication, authorization, storing of experiment log or result, etc. Based on the experimental specification given by the user appropriate code is loaded on the Freescale Semiconductor kit the results are displayed in the Digital Storage Oscilloscope connected to it. These results are sent to service broker to store it in database and through live streaming option the execution process can be displayed on client machine.

4. Lab Client Design of ARL

In figure 3, the ARL lab client interface is shown. It includes three important components namely user interface, core module, and server interface.

User interface provides a front end window to access various functions such as submission of experimental specifications, validation of parameters before submission, lab configuration information, lab information URL, retrieving results, etc. Core module, is the lab client of ARL, and it sends the user request to the server through an interface. Interface can pass parameters to service broker in order to store and conduct the experiment.

A user logs into service broker website with username and password. Service broker enumerates user’s authorized groups. Then user selects an effective group (one of the authorized groups). Service broker enumerates Lab Clients available for the user. User loads a lab client. Client reads user’s preferences data from the data store. The client calls Get Lab Configuration method for configuration settings for a specific experiment. User creates experimental specifications. Client calls Verify() before submitting it to the server. Client calls Submit() method. After submitting specification to the server, the user needs not to stay online. Once the experiment is completed, the client calls Retrieve-Result() method. Client displays the results to the user. User closes the client and logs off from the Service Broker.
5. Design of ARL Lab Server

Figure 4 shows the components such as lab web site, web service API, interface to lab device. The experiments should be executed through automated package or engine called Experimental Execution Engine. Frequency Generation is an experiment to generate frequency on the Digital Storage Oscilloscope, from client end. The user needs to give the input specification to execute the experiment. Before sending the specification to the LS, specification values are sent for validation. If the specification is valid then only it is sent to the LS for execution and the results are sent back to LS, using some interface [10].

![Figure 4](image)

**Figure 4. Components of ARL Lab Server**

6. Steps to Access Laboratory Experiment

In figure 5, we explain how to access Web-Enabled Remote Laboratories experiment [11]. In table 1, steps to access the lab experiment are explained.

![Figure 5](image)

**Figure 5. Steps involved in conducting laboratory experiment**

The frequency generation experiment uses Freescale Semiconductor kit namely the HCS12DP256B microcontroller and for the pulse width modulation experiment we use the HCS12DTX512 microcontroller in order to generate the frequency in the DSO. Through this kit many embedded systems experiments can be conducted. The application loads the appropriate code into the erasable memory of the kit. After loading the code into the kit, as per the memory location specified in the code, the user supplied values are stored on a particular memory and the execution starts. All the said process and steps are automated in order to provide lab facility from a remote machine at any time. From anywhere the experimental execution process can be observed using streaming media services.

### Table 1. Interaction and Web Services Calls between Lab Client, SERVICE BROKER and Lab Server

<table>
<thead>
<tr>
<th>Step</th>
<th>Student Activity</th>
<th>Service Broker Activity</th>
<th>Lab Server Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Authenticates over SSL</td>
<td>← Lists student’s groups (classes)</td>
<td>→ Lists available Lab Clients (experiments)</td>
</tr>
<tr>
<td>2.</td>
<td>Choose group (class) for session</td>
<td>← Lists available Lab Clients (experiments)</td>
<td>→ Launches Lab Client</td>
</tr>
<tr>
<td>3.</td>
<td>Choose Lab Client (experiment)</td>
<td>← Launches Lab Client</td>
<td>→ Calls WS Submit (experimentSpecification)</td>
</tr>
</tbody>
</table>

---

168
11. Executes experiment and ← Calls WS NOTIFY(EXPERIMENTID)

12. Calls WS → RETRIEVERESULT (EXPERIMENTID)

13. ← RETURNS RESULTREPORT

14. Calls WS → RetrieveResult( experimentID)

15. ← Returns ResultReport

In this table, → denotes information or data flow from left side source component to right side destination component. ← denotes information flow from right side source component to left side destination component.

1) The student starts the session by logging in to the Service Broker (SB) using a secure connection (SSL). The SB provides an implementation of a standard name and password authentication scheme.

2) SB responds by displaying a list of the groups in which the student is registered. Groups usually correspond to classes.

3) The student selects one of the available groups (classes) for his session.

4) The SB displays all the available Lab Clients belonging to the group. A Lab Client usually corresponds to a single experiment.

5) The student selects one of the available Lab Clients.

6) The SB now launches the Lab Client.

7) This marks the transition in the student’s session from communicating with the SB using a web browser to view the SB’s active server pages to the running of an experiment during which the student communicates using the Lab Client and web services. We have used the client technology called HTML Client. The student edits the description of the experiment to be run using this client.

8) When the experimental specifications is complete, the student directs the client to invoke the web service Submit() method on the Service Broker. Submit() takes a text encoded version of the experimental specifications as an argument. The SB is not expected to understand the specification. The SB stores a copy of the experimental specifications and forwards the Submit() call to the LS.

9) The LS immediately returns a submission report that includes any error messages resulting from illegal experimental specifications. If the specification is legal, the LS queues the experiment for execution.

10) The SB forwards the submission report back to the client along with an integer experiment ID that all parties now use to identify the experiment.

11) Once the LS executes the experiment, the LS calls the Notify() web service on the SB to indicate that the experiment’s results are now available.

12) The SB now requests the results from the LS using the RetrieveResult() web service.

13) The LS returns the results and any error messages to the SB, which stores but it can not interpret the experimental results.

14) The client can request the results (may be in XML file format or to visualize the execution of the experiment based on the supplied input parameters he can lively watch the execution process using streaming services) from the SB.
using the client’s the RetrieveResult() web service.

15) The SB returns the results and any error messages.

7. Results

![Figure 6. Frequency Generated on DSO for 112.08 microseconds of ON time and 130.20 microseconds of OFF time](image)

In figure 6 above, a sample result image is shown for the input experimental specification value i.e., ON time of 4 and OFF time of 5 for the frequency generation experiment. This experimental specification can be forwarded to Service Broker, SB can validates it, if these values are valid, an appropriate program can be loaded into Freescale Semiconductor kit – derivative HCS12DP256B using Freescale CodeWarrior IDE software package and hardware interface, loaded code is executed by the given experimental specification in appropriate memory location of the 16 bit microcontroller kit. The frequency is generated on the DSO and the results are sent back to LS for the calculation of frequency value.

8. Conclusions and Future Works

Web-Enabled Remote Laboratories is a collaborative system that allows students to do their lab works from anywhere at anytime by any number of times. This system will avoid damages to devices. Instead it conducts the experiment on real device in an automated manner. i.e., no human intervention is required on setting up the device to execute experiment. It requires only the specification or input parameters for the specific experiment [12]. Everything is shared across the LC, SB, LS by establishing a web service communication via SSL connection. The credentials and passkey details are stored in the SOAP header of XML file and are shared across the system via SSL connection. This system is more secure than any other system. The whole system has been developed using Web Service framework so that the proposed system is platform or domain independent. As a future work, using the same architecture some of the other experiments can be made accessible from anywhere anytime through an Internet / Intranet.

Acknowledgements

This work forms part of the TIFAC-CORE in “Automotive Infotronics” project. The instruments and all other resources used for the development of this project “Accessing Real Labs using Web Services” are provided by Freescale Excellency Lab for Automotive, TIFAC-CORE, VIT University, Vellore, India. All the Application Programmable Interfaces (APIs) are used in Service Broker module, are provided by Massachusetts Institute of Technology, Cambridge, USA.

References


