Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics
San Antonio, TX, USA - October 2009

Design and Implementation of a Service-Oriented Web-based Control Laboratory

Yuliang Qiao, Guo-Ping Liu, *Senior Member, IEEE*, Geng Zheng

Abstract—**This paper introduces the NCSLab (Networked Control System Laboratory) at http://www.ncslab.net, which provides a complete Web-based solution for users to carry out experiments on experiment devices located globally. A scalable and service-oriented architecture which is composed of Web browsers, central Web server, MATLAB servers, regional experiment servers, control units and experiment devices is proposed. Based on the architecture, many novel functionalities including visual algorithm designing, simulation, compilation, visual monitor configuration , real-time monitoring and supervisory control are designed and implemented by combination of state-of-the-art technologies such as Web2.0, J2EE and MATLAB. The service-oriented architecture designed and the functionalities provided distinguish NCSLab from other existing remote laboratory systems. An experiment example is given to demonstrate that users can enjoy all these rich interactive features with a simple Web browser from anywhere at any time.**

I. Introduction

Remote laboratories are real equipment laboratories which enable users to operate real devices via computer network and thus conduct experiments remotely. Therefore, lab resources can be shared among different universities. Remote learners or researchers can access remote laboratories from anywhere at any time, and obtain practical experience from experiments in remote laboratories.

There exist some solutions of remote laboratories. Among them, some are restricted to a particular experiment device (DC motor[1], inverted pendulum [2], robot [3,4] and 2-DOF helicopter [5] for example; some solutions require users to download a special software, such as Java Run-time Environment [2] or even a client terminal [1], which is often not favored by cautious Web users. As to the functionalities provided by remote laboratories, some only enable users to run predefined programs and monitor the results [6], some allow users to adjust given parameters and watch how the real plant/process responses [7], and others give users the freedom of designing their own controller algorithms or reference inputs [8].

This work is supported by National Science Foundation of China under Grant 60528002, 60621001 and the National High Technology Research and Development Program of China (863 Program) under Grant 2007AA04Z202.

Yuliang Qiao and Geng Zheng is with Institute of Automation, Chinese Academy of Sciences, Beijing, China e-mail: aqiaojoe@gmail.com)
G.P. Liu is with the Faculty of Advanced Technology, University of Glamorgan, UK, and is also with the CTGT Center in Harbin Institute of Technology, China.

In this paper, NCSLab, a remote control laboratory, which aims at integrating various experiment devices located globally and offering experiment service to users scattering around the world, is presented. Furthermore, NCSLab also intends to free users from the constraints of time, space, software environments of their PC (operation platforms and Web browser, to be specific), programming knowledge mastered and provide them with a wide range of interactive experience from designing a new control algorithm to customizing their own monitoring during experiments.

This paper is organized as follows: Section 2 describes the architecture of NCSLab; Section 3 explores the design of integrated Web-based solutions and their technical implementation, including algorithm design, simulation, compilation, experiment monitor configuration, real–time monitoring and supervisory control; Section 4 illustrates an experiment session in NCSLab. The last section summarizes this paper and suggests the possible research directions of NCSLab.

II. Architecture of NCSLab

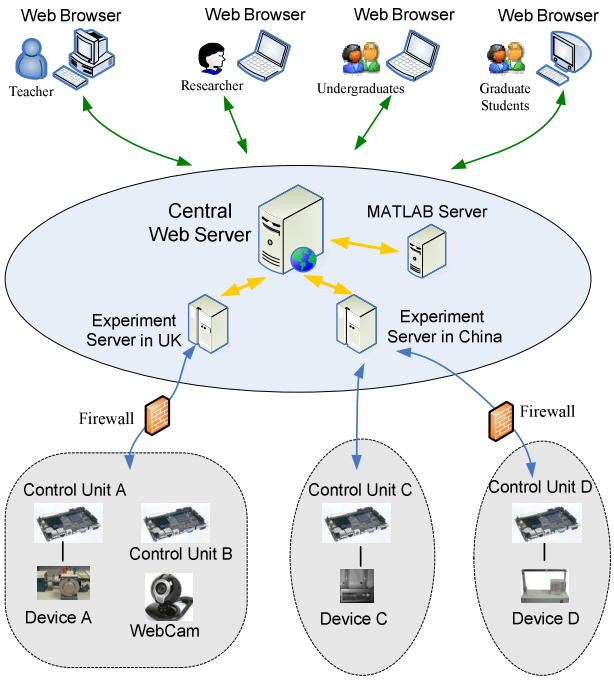

Fig. 1 the architecture of NCSLab

Considering that NCSLab is required to connect the globally located experiment devices and provide experiment services to users whose Web browsers are dispersed in the

978-1-4244-2794-9/09/$25.00 ©2009 IEEE

environment of Internet, this paper proposes and designs a scalable architecture for NCSLab. From a top-down view, the architecture consists of Web browsers, central Web server, MATLAB servers, regional experiment servers, device control units and experiment devices, as is shown in Fig. 1.

1. *Web Browsers*

A Web browser is assumed to be the only software environment needed by users of NCSLab, and it is often available for a modern personal computer. HTML, JavaScript/ AJAX and Flash technology are utilized to implement the interactive functionalities, and they are supported by almost all the prevalent Web Browsers such as Internet Explorer, Mozilla Firefox, Safari, Netscape or Opera etc. The feathers of zero client installation and cross-browsers of NCSLab increase its availability and promote users' experience.

2. *Server Cluster*

The server side of NCSLab is composed of a cluster of distributed servers to support a better scalability and real-time performance. They communicate with each other using J2EE technology such as remote method invocation (RMI).

2.1 Central Web Server

The central Web server resides in the top of the architecture except Web browsers, and it is the core of NCSLab. It manages the global resources of NCSLab such as experiment server registry, MATLAB server registry, experiment device registry and user registry etc., and the data of these resources are stored persistently in a MySQL database server. In addition, it generates the user interaction interfaces in the form of Web pages and communicates with Web browsers via HTTP protocol. The famous open-source software Tomcat released under the Apache Software License is selected to implement the Web server.

2.2 MATLAB Servers

A MATLAB server is an application server that provides service of remote simulation and remote compilation for the central Web server, and thus users can perform web-based simulation and compilation via the central Web server without installing MATLAB on their local computers. NCSLab adopts MATLAB as the modeling and simulation kernel because MATLAB is very popular in academia and industry, so users can learn to design an experiment in NCSLab rapidly with MATLAB's experience.

2.3 Regional Experiment Servers

Regional experiment servers are located in Internet environment, just like the central Web server. A regional experiment server provides experiment services such as starting experiment and stopping experiment for the central Web server. Furthermore, once an experiment is started, the regional server that is closest to the experiment device is chosen to communicate with the device and provide real-time monitoring service to Web users during the process of experiment. The number of regional servers can be expanded according to the number and network distribution of experiment devices all over the world; for the present there are two regional servers deployed in the current NCSLab: One in UK and the other in China.

3. *Device Control Units*

Device control units are physically connected to experiment devices via cables. The hardware environment of control units is mainly based on a high-performance 32-bit ARM RISC microprocessor. It is composed of a main board, an AD/DA board, a LCD board and an I/O board [9]. The solution is very cost-effective for computer-controlled systems.

Based on the hardware, a concise Linux operation system is running in the control unit. In addition, device I/O driver modules, PWM driver modules, network driver modules can be loaded whenever needed. The device I/O and PWM modules enable a control unit to collect sensor signals from its controlled device and drive the device, and the network driver module enable it to communicate with other hosts or control units in the network. These driver modules can be accessed by calling some C subroutines programmed by developers of device level.

4. *Experiment devices*

For an experiment device to be incorporated into NCSLab, it must be able to be driven by electrical signals such as voltage or current, which is often possible for modern experiment devices. Furthermore, it must be clearly defined which input signals can be tunable and which output signals can be measured. By doing so, an appropriate model for the experiment device can be built, which is essential for the design of controller. There are many kinds of devices connected to NCSLab, such as the servo motor, water tank, ball-beam, magnetic levitation and so on.

It can be seen from above that the architecture of NCSLab service-oriented, scalable and secure; furthermore, compared with the architecture reported in [10], it reduces network delay during remote experiment dramatically because the experiment server that is closest to the device can be chosen to communicate directly with control units and send real-time data directly to the users' browser, without passing through a sub server and accessing some database servers.

III. DESIGN AND IMPLEMENTATION OF NCSLAB

Based on the above architecture, NCSLab offers powerful functionalities and rich features, which render users a whole Web-based control experiment solution from algorithm designing, simulation, compilation, monitoring configuration to real-time monitoring and supervisory control. Users can complete the whole process rapidly with just a normal Web-browser and they can enjoy desktop-like interactive experience. This section will present the design and implementation of these functionalities.

1. Web-based Designing , Simulation and Compilation

NCSLab views every experiment device as a model that has known specifications of inputs and outputs, and provides a pair of models for each device in the MATLAB server. One is the simulated model, which combines some Simulink blocks to simulate the physical characteristics of the device and is used in simulation control; and the other is the physical model, which includes S-function blocks that can access the physical inputs and outputs of experiment devices (I/O and PWM etc.) after compilation and is prepared for practical control. The

two models expose the same inputs and outputs interfaces to its outside, so a control diagram designed for simulation can be reused in practical control by only replaying the part of the controlled device, as is shown in Fig. 2 . This replacement is made by MATLAB server automatically, and therefore experiments can be transited seamlessly from simulation to practical control.

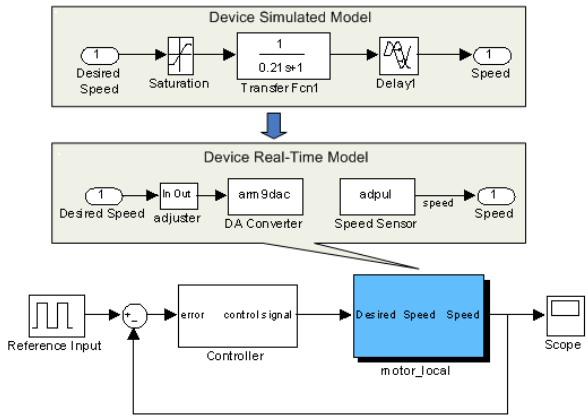

Fig. 2 Replacement from the simulated model to the real-time model for a DC motor in Chinese Academy of Sciences

1.1 Web-based designing of control scheme

NCSLab provides some pre-defined control algorithms for every experiment device, and users can carry out experiments using them. They can also design their own algorithms in a Web-based graphic user interface (GUI).

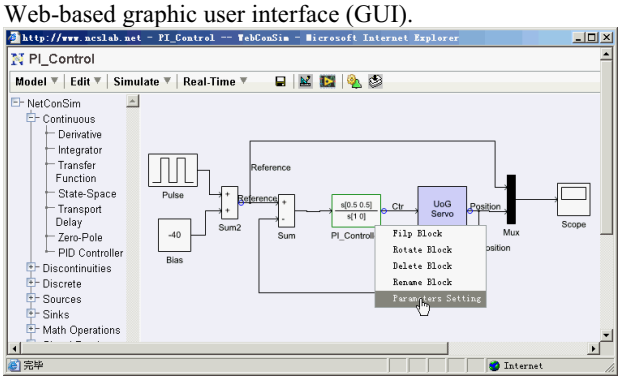

Fig. 3 the Web-based GUI where users can design control algorithm, do simulation and compilation (shown in Microsoft Internet Explorer 6).

The GUI designed is characteristic of What You See Is What You Get (WYSIWYG), as shown in Fig. 3. To design a control scheme, users just need to draw a control diagram by clicking-and-dragging, and they can get nearly the same experience as using Simulink. They can drag a block from the left block-library tree; they can draw lines to connect the output port of a block to the input ports of other blocks. If they double-click a block or a line, a dialog window will pop up where they can configure its parameters. This visual designing method is very intuitive, and users do not need to have knowledge of any programming language. The GUI just uses Web browsers' built-in functionality (such as VML for Microsoft Internet Explorer and SVG for Moniza Firefox) to support drawing graphics, so users can enjoy the graphic designing experience without installing any browser plug-in.

Users' actions in GUI are responded by local Web browser, instead of the remote Web server. This ensures the timely interaction and improves the users' experience. Users' control diagram data are stored temporally in the memory of local browser and are organized in the format of JSON (JavaScript Object Notation).

1.2 Web-based simulation

In fact, the real simulation environment resides in the MATLAB server, and the simulation functionality of MATLAB is wrapped as a service via JAVA RMI (Remote Method Invocation) technology. After the central Web server looking up the registry of a MATLAB server, it can remotely call this service via the help of Remote Simulation Invocation Engine, which implements a RMI stub client.

When a user finishes designing a control diagram and clicks the "Simulate" button, the JSON representation of his/her diagram is submitted to central Web server. The central Web server launches the remote simulation invocation engine and includes the JSON data as the input of the remote invocation. MATLAB Server then parses the JSON data and translates them into a Simulink model file. Of course, the model of the controlled device in the file is linked to be the model in device simulated library, as described above. MATLAB Server then to do simulation based on the model file using Simulink and generates the simulation results files, including the textual data and graphic curves. The results are then sent to the central Web server as the return value of the remote compilation invocation. At last the graphic curves and data are shown in user's Web browser and can be downloaded from the Web page for an off-line analysis. The whole process of Web-based simulation is shown in Fig. 4.

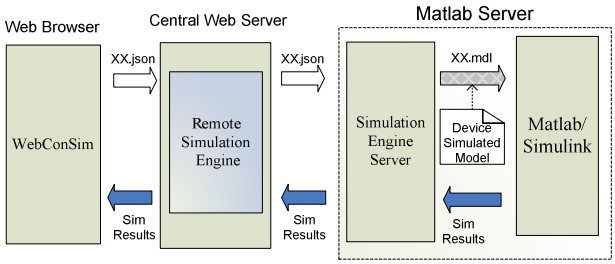

Fig. 4 the Flowchart of Web-based Simulation

1.3 Web-based compilation

The Simulink model needs to be compiled into real-time executable codes before real-time experiment. The compilation process is divided into two parts. First, RTW toolbox generates C codes from Simulink model. Then, the C codes are complied into real-time executable codes by ARM-Linux GCC Compiler. In view of the platform difference between a MATLAB server deployed on the Windows-NT and a control unit deployed on ARM-Linux, CygWin is used to build a cross-compilation environment in the MATLAB server and enable the complied codes to be executable on the ARM-Linux OS in a control unit.

Fig. 5 shows the flow of Web-based compilation. It should be pointed out that in the compilation process the MATLAB server is configured to select the model of the controlled device from the device real-time model library, and thus the compiled executable codes has got the ability to interact with

physical device. At last, the generated executable codes are returned to Web server as the result of remote complication invocation and stored in the algorithm registry of the central Web server for later utilization.

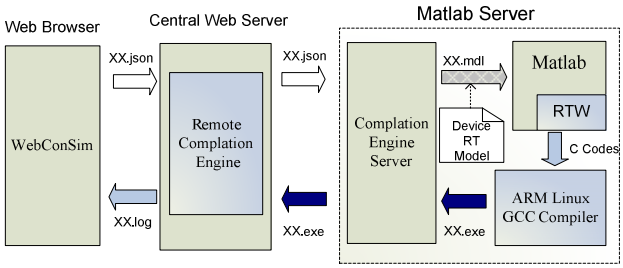

Fig. 5 the Flowchart of Web-based Compilation

Although the backstage mechanization of Web-based compilation is very complicated, the processes are executed consecutively and intelligently on the server side, and they are transparent to users. On the side of the browser, the User just needs to click a "Compile" button and wait for the results. So users can concentrate themselves on the design of control algorithms without caring about how to enable the algorithms to run on the practical experiment.

2. Web-based Experiment

A real-time experiment is started as soon as an algorithm is downloaded into a control unit from the Central Web server, which action can be invoked by a user's clicking on the Web page interface which is related to the algorithm. In fact, the real-time algorithm runs as a separate computer process in the control unit. Besides controlling the physical plant, the control unit automatically launches a SCADA server at the same time. This server can watch the running states of the algorithm and thus provide built-in monitoring service. Remote network nodes can get this service as long as they follow the Netcon Supervisory Protocol (NSP) [11]. This protocol is built upon the prevalent TCP/IP protocol and implemented in the NCSLab using C Socket programming.

2.1 Visual Configuration of Monitoring Interface

Besides allowing users to design their own control algorithms, NCSLab also gives users the freedom to customize their own monitor interface, in which they can decide which signals to be observed and which parameters to be tuned during experiments. This is a creative solution which the other remote laboratories do not provide.

The experiment execution engine of experiment server and control units exchange information via NSP protocol. As soon as an algorithm experiment is started, the engine connects to the SCADA service port of control unit, send an EXT_GET_PARAMSIGNAL message, and obtain a list of all the tunable parameters and observable signals. The list of parameters and signals is then stored in the server for users' later selection.

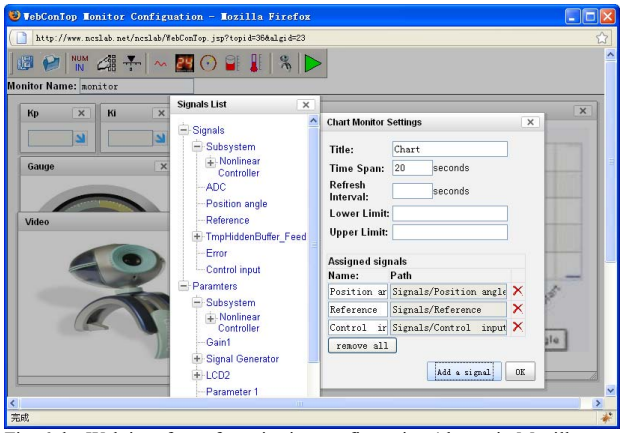

Fig. 6 the Web interface of monitoring configuration (shown in Mozilla Firefox 3)

To support more friendly use, NCSLab developers have designed the configuration page to be visualized and a set of virtual instrumentation are implemented in the configuration page. They include the components of Numeric Input, Slider Input, Real-time Chart, Angular Gauge, Cylinder Gauge, Thermometer and Video Monitor. Users can setup a fresh and colorful configuration just by clicking and dragging these virtual instruments. They can double-click the virtual instruments to select signals or parameters they are interested in for the instrument from the tree list which includes all the parameters and signals, and their configuration can be saved on the central Web server for a second use.

2.2 Web-based Monitoring and Supervisory Control

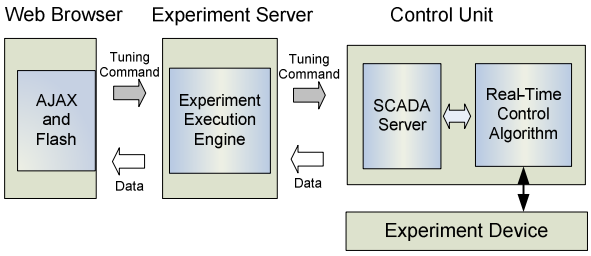

Fig. 7 the Flowchart of Web-based monitoring and tuning

When the user finishes configuration of monitoring interface, he/she can move on to do real-time monitoring and supervisory control. Fig. 7 depicts how the information flows in the process of remote monitoring and tuning.

The user who gets the full control right can tune some parameters of the real-time algorithm, and his/her tuning commands are received by the regional experiment server. The experiment engine then encapsulates the parameter name together with its new value into an EXT_SETPARAM message, and sends the message to the control unit, which adjust immediately the specified parameter of the algorithm according to the body content of the message; the control unit also sends EXT_UPLOAD_DATA messages continuously which contains the latest real-time data of all selected signals. The newest data are immediately received by the regional server and stored in a buffer of memory. Web browsers can acquire the up-to-date data automatically from the server by

and refresh the monitoring interface with the help of Web 2.0[12] technology such as AJAX and Flashes.

IV. Experiment Example

This section will demonstrate an example to illustrate how to carry out experiments on NCSLab Website. In the example, a user located in Chinese Academy of Sciences selects the servo motor control test rig in University of Glamorgan, UK as the experiment device. The control objective is to control the angel of the position plate tracking the preset angle, and he/she wants to complete the whole process from simulation to real-time supervisory control during the experiment.

Following the instruction document of NCSLab, the user logins NCSLab and navigates to the experiment Web page which belongs to the test rig. The user notices that a model for the controlled device has been created by the device manager. The model uses a setting speed as input and the response speed as output, its transfer function representation is identified approximately as:

$$G(s)=\frac{0.03868s^3-5.369s^2-97.34s+13450}{s^3+169.8s^2+1730s+159}$$

Based on the model, the user designs an on-line control diagram named "TF_Control". The diagram uses a transfer function as the controller and gives a square-wave signal as the reference input, as depicted in Fig. 8.

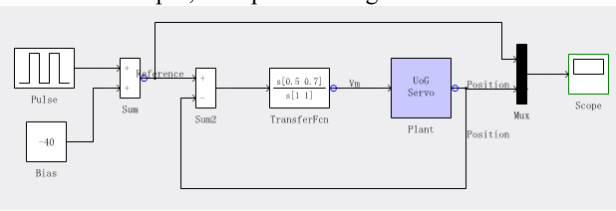

Fig. 8 the on-line transfer function control diagram

Then the user clicks the "Simulate" button and a panel pops up to show the simulation progress. After a few seconds, the simulation is completed at the remote server and the graphic result is displayed on the pop-up panel in the local Web browser, as shown in Fig 11. The user can return to the design panel and modify controller parameters until he/she obtains a satisfactory simulation result.

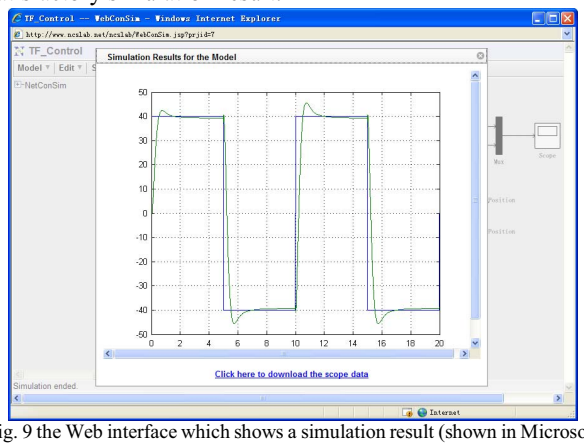

Fig. 9 the Web interface which shows a simulation result (shown in Microsoft Internet Explorer 7)

After simulation, the user clicks the "Compile" button and the previous panel will pop up again to show the progress of remote compilation. The real-time codes of his algorithm are generated in several seconds and the hyperlink to the algorithm is displayed immediately. The user then navigates to the page of the algorithm after clicking the hyperlink.

The user then requests the full control of the device. It is assumed that there are no other users who compete for controlling the same device, so he/she obtains the control right immediately. After he/she clicks the "Start Experiment" button, the real-time algorithm codes are downloaded to the control unit automatically, which starts the practical control

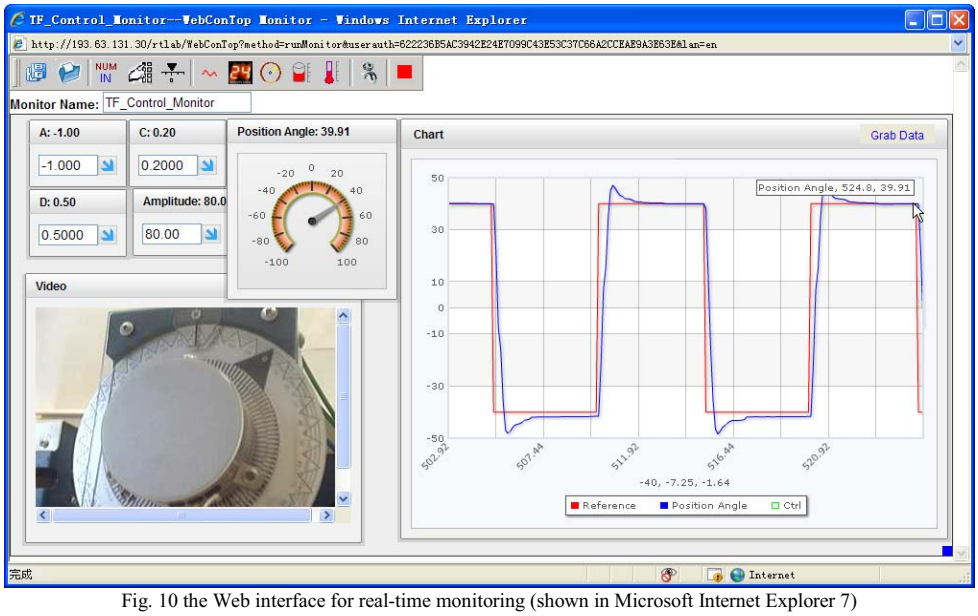

Fig. 10 the Web interface for real-time monitoring (shown in Microsoft Internet Explorer 7)

experiment.

Then, the user begins to configure a supervisory interface. He/She drags a video widget to watch the on-the-spot scene, a chart widget to display the curve of the actual angle position together with the reset position and some numeric input widgets for controller parameters.

At last, the user clicks the "Run Monitor" button, and then the real-time monitoring GUI is displayed, as shown in Fig. 10. The widgets configured just now begin to work and automatically fetch the latest data/images and refresh themselves. Due to the uncertainty of the device model, the practical response is not totally the same with the simulation result; so the controller parameters can be tuned until the servo control system reaches the control objective both in static error and dynamic responses. At last, the user can click the "Grab Data" button to save the experiment results on his local computer for further study.

V. CONCLUSION

In this paper, NCSLab, the Web-based remote laboratory which provides users with rich interactive features is presented. A scalable architecture is proposed and a complete Web-based solution from visual algorithm designing, simulation, compilation, monitoring configuration to real-time monitoring and supervisory control are designed and implemented. Users can thus carry out experiments in NCSLab without installing any software and mastering any complicated programming language. So NCSLab can be used to test fresh ideas in academic research or enhance practical knowledge in e-learning in the fields of automatic control system, and it can also be extended to other engineering fields.

Although NCSLab is sophisticated and easy to use, its effects on control education and research are to be evaluated. In addition, it is also a challenge for NCSLab to keep track of the ever-evolving network and Web technology and benefit from them in order to become more powerful, friendly and secure. So future work on NCSLab can be devoted to the following directions:

a. To introduce experiment collaboration and competition of control performance between users to NCSLab.

b. To publish experiments on NCSLab as a Web service so that other kinds of remote laboratories can find and use them conveniently.

c. To win cooperation from more institutes and universities so that they can share their lab resources.

REFERENCES

[1] T. Kikuchi, T. Kenjo and S. Fukuda, "Remote laboratory for a brushless DC motor," IEEE Transactions on Education, vol. 44, no. 2, pp. 12, 2001

[2] J. Sanchez, S. Dormido, R. Pastor, and F. Morilla, "A Java/MATLAB based environment for remote control system laboratories: Illustrated with an inverted pendulum," IEEE Transactions on Education., vol. 47, no. 3, pp. 321–329, Aug. 2004.

[3] Min Wu, Jin-Hua She,Gui-Xiu Zeng and Ohyama, Y. ,""Internet-Based Teaching and Experiment System for Control Engineering Course", IEEE Transactions on Industrial Electronics, June 2008, vol. 55, no. 6, pp. 2386-2396

[4] C. S. Tzafestas, N. Palaiologou, and M. Alifragis, "Virtual and remote robotic laboratory: Comparative experimental evaluation," IEEE Transactions on Education, vol. 49, no. 3, pp. 360–369, Aug. 2006.

[5] C. C. Ko, B. M. Chen, J. P. Chen, J. Zhang, and K. C. Tan, "A Web-based laboratory on control of a two-degrees-of-freedom helicopter," Int. J. Eng. Educ., vol. 21, no. 6, pp. 1017–1030, 2005.

[6] S. Qin, L. Bo and X. Liu, "Development of the networked virtual instrument lab for vibration measuring based on Microsoft.NET," Proceeding of the 21st IEEE Instrumentation and Measurement Technology Conference, vol. 2, pp. 1286-1289, 2004

[7] Hercog, D., Gergic, B., Uran, S., Jezernik, K., "A DSP-Based Remote Control Laboratory," IEEE Trans. On Industrial Electronics, vol. 54, no. 6, pp. 3057 – 3068, Dec. 2007.

[8] M. Casini, D. Prattichizzo and A. Vicino, "The Automatic Control Telelab: A Web-based Technology for Distance Learning," IEEE Control System Magazine, vol 24, pp. 36-44, 2004.

[9] Bian Haibo, Liu G. P., Dong, Zhe, Structure Design and Application of Embedded Ethernet Based Control Systems, Proc. of 2007 IEEE International Conference on Networking, Sensing and Control, pp. 47 – 51.

[10] Hu W. S., Liu G. P.; Rees D.; Qiao Y. L., "Design and Implementation of Web-based Control Laboratory for Experiment devices in Geographically Diverse Locations", IEEE Transactions on Industrial Electronics, June 2008, vol. 55, no. 6, pp.2343-2354

[11] Zhu Youzhi, Zheng Geng, Dong Zhe, Liu, G.P., Design of Supervisory Software for Ethernet-Based Control Systems, Proc. of 2007 IEEE International Conference on Networking, Sensing and Control , pp. 36-41.

[12] O'Reilly T. "What Is Web 2.0", September 2005, http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-Web-20.html