AC 2011-2796: DEVELOPMENT OF A MODULARIZED ARCHITECTURE FOR REMOTE-ACCESS LABORATORIES

El-Sayed Aziz, Stevens Institute of Technology

Dr. El-Sayed Aziz holds a faculty position as assistant professor in the Production Engineering and Mechanical Design Department at Faculty of Engineering, Mansoura University, Egypt. Currently, he is working as research scientist at Stevens Institute of Technology, Hoboken, New Jersey, USA. He received B.S. and M.S. degrees in Mechanical Engineering from Mansoura University, Egypt, in 1991 and a Ph.D. in Mechanical Engineering from Stevens Institute of Technology in 2003. His research interests include knowledge-based engineering systems, computer-integrated design and manufacturing, Finite Element Analysis, software development and applications as well as remote and virtual laboratories.

Zengqian Wang, Stevens Institute of Technology
Dr. Sven K. Esche, Stevens Institute of Technology

Sven Esche is a tenured Associate Professor at the Department of Mechanical Engineering at Stevens Institute of Technology. He received a Diploma in Applied Mechanics in 1989 from Chemnitz University of Technology, Germany, and was awarded M.S. and Ph.D. degrees from the Department of Mechanical Engineering at The Ohio State University in 1994 and 1997, respectively. He teaches both undergraduate and graduate courses related to mechanisms and machine dynamics, integrated product development, solid mechanics and plasticity theory, structural design and analysis, engineering analysis and finite element methods and has interests in remote laboratories, project-based learning and student learning assessment. His research is in the areas of remote sensing and control with applications to remote experimentation as well as modeling of microstructure changes in metal forming processes. He publishes regularly in peer-reviewed conference proceedings and scientific journals. At the 2006 ASEE Annual Conference and Exposition in Chicago, USA, he received the Best Paper Award for his article 'A Virtual Laboratory on Fluid Mechanics'.

Constantin Chassapis, Stevens Institute of Technology

Development of a Modularized Architecture for Remote-Access Laboratories

Abstract

Conducting hands-on experiments in undergraduate laboratory courses with large student enrollment imposes significant strains on the fiscal, spatial and personnel resources of the educational institutions. In response to the need for developing laboratory resources that provide a practical experience to large engineering classes, remotely shared experimental facilities have emerged as one innovative solution for educational laboratories with reduced resource needs. Recent research shows that a significant number of remotely-accessible experiments have been deployed globally across many of the major engineering disciplines around the world. Several software architectures and technologies for remote laboratories have been proposed and implemented over the last years.[1] Organizations usually choose and adopt one solution based on their needs, previous experience, available software and software development tools as well as the skills and expertise of the developers. Each solution has its advantages and disadvantages. In this context, there is an increasing need for a unified method for developing and presenting such remote-access laboratory resources in order to allow potential users to easily and efficiently use them.

The aim of this paper is to present a modularized and scalable system architecture for remote experimentation, which enables the interaction of multiple users with a network of spatially distributed experimental devices. Furthermore, this paper describes the functionality available in the current version of the remote-access laboratory that enables students to run a wide range of experiments on this platform. As an example, the implementation and some experimental results for a remotely accessible wind tunnel are presented, including detailed descriptions of the techniques employed for linking the different functional modules implemented as LabVIEW scripts to an online laboratory system. This modularized remote laboratory system was designed based on a client-server structure. The wind tunnel setup enables the students to explore various fluid flow phenomena, such as the external air flow around an airfoil and a golf ball.

1. Introduction

Modular programming[2] is a term used to describe the subdividing of a program into separate subprograms such as functions and subroutines. It represents a software design technique that increases the extent to which software is composed of separate, interchangeable components, called modules. Conceptually, modules[3] represent a separation of concerns, and they improve maintainability by enforcing logical boundaries between components. With modular programming, concerns are separated such that no (or few) modules depend upon other modules of the system, and having as few dependencies as possible is the goal. Modules[4] are typically incorporated into the program through interfaces. In the field of computer science, an interface refers to a point of interaction between multiple components, which allows a component, in this case a subprogram, to function independently while using interfaces to communicate with other components via an input/output system and an associated protocol. A module interface expresses the elements that are required and provided by the module. The elements defined in the interface are detectable by other modules. The implementation contains the working code that corresponds to the elements declared in the interface.

When creating a modular system, instead of creating a monolithic application where the smallest component is the whole application, several smaller modules are built (and usually compiled) separately, which – when composed together – construct the executable application program. If realized correctly, this makes systems designed in a modular fashion far more reusable than those based on the traditional monolithic design, i.e. all (or many) of these modules may then be reused without change in other projects. This approach also facilitates the 'breaking down' of projects into several smaller projects. Theoretically, a modularized software is more easily assembled by large teams, since none of the team members are creating the whole system, or even need to know about the system as a whole. Instead, they can focus just on the assigned smaller task.

Modular programming is widely used in software projects. Therefore, it is possible to concentrate on the actual logic of the application while reusing the infrastructure, frameworks and libraries written and provided by others. The modular programming technique enables the independent debugging of the pieces by dividing the work amongst multiple programmers, which greatly reduces the development time and effort. Furthermore, it makes code reusable, thus rendering the whole project easier and more logical. Subprogram modules also make it simpler to understand how the program operates. If for instance the boundary conditions are implemented using a subroutine, the program can be searched for this subroutine to find all places where these boundary conditions are used. This might include some unexpected places, such as in the output or in performing a numerical check on the overall accuracy of the program. Furthermore, the usage of subprograms tends to shorten the whole program in terms of lines of code compared to monolithic programs. If the subprograms have no errors, debugging is simplified to the communication and protocols between the modules. Meanwhile, modularization reduces the likelihood of programming errors in general. Since subprograms can use local variables, there is less of a chance that the code in the subroutines interferes with that of the main program or other subprograms.

Remote laboratories have been developed and implemented in various science and engineering departments worldwide. Based on the existing remote laboratories at Stevens Institute of Technology (SIT)[5,6], developments at other institutions and a study of modular programming, this paper presents a modularized approach for developing remote experimental setups for different engineering and science disciplines. Then, this modular programming concept is applied to develop remote-access to a wind tunnel for an undergraduate fluid mechanics course. The modular system design and implementation and the communication between modules for general cases are discussed.

2. Overall structure of remote-access laboratories

A remote-access laboratory platform involves a real, remotely located, physical system (i.e. an experimental device together with actuators and sensors to manipulate and monitor it), in conjunction with visual and data feedback from the remote site (i.e. some type of 'real presence' at the remote site). In the field of engineering education, although traditional hands-on laboratories are effective and vital for illustrating theoretical concepts taught in lectures, the significant space, time and personnel costs imposed on the educational institutions by these traditional laboratories have spurred new ideas for conducting engineering experiments. The benefits of remote-access laboratories are predominantly in engineering education.[7] Remote laboratories share the advantage with simulation-based laboratories that they require minimal space and that the experiments can be rapidly configured and run over the Internet. In addition, remote laboratories provide real data, which are subjected to all possible uncertainties affecting the real equipment. Meanwhile, sharing laboratories by large numbers of users leads to a significant reduction of the large fixed costs of traditional laboratories, thus leading to economies of scale. With remote laboratories, students are able to repeat experiments, for instance to clarify doubtful measurements, and therefore, the quality and effectiveness of experiments are improved significantly. Furthermore, since there is no risk of catastrophic failure, safety and security are improved.[8]

Starting in the 1990s, many remote laboratories[9,10,11] have emerged all over the world. At SIT, a remote laboratory was implemented using a client-server network architecture, which enables the concurrent execution of multiple experiments using separate experimental setups.[12,13] Experiments involving the same experimental setup are queued and executed in the order of the incoming requests. Based on this architecture, different experimental setups for dynamical systems were implemented, such as a one-degree-of-freedom mechanical vibration system, a duct acoustic system, a liquid-level system and various electrical systems. An innovative real-time remote-access control engineering teaching laboratory[14] was developed and demonstrated at Oregon State University in 1998. A remote laboratory called VLAB[15] involving an oscilloscope was set up at The National University of Singapore in 1999. Later, a Web-based experiment[16] for controlling a coupled tank apparatus was developed. In the Process Control and Automation Laboratory[17] at Case Western Reserve University, a process rig was made accessible over the Internet, where the user can submit parameters using a Web browser from a remote client to a Laboratory Virtual Instrument Engineering Workbench (LabVIEW) Web server, which is connected to the process rig via a PLC control module. An interactive online laboratory for remote education called Automated Internet Measurement Laboratory[18] was established at Rensselaer Polytechnic Institute. A course module on semiconductor device characterization was developed, which could be accessed through a Web browser. Some of the remote experiments are conducted in real time, controlled remotely by either webpage plug-ins using LabVIEW's Web serving facility or by separate LabVIEW remote client programs.[19] Other experiments are run in batch mode, where the experiment specifications are submitted by the users through a Microsoft Internet Information Server. The characteristics and basic ideas of these two methods are discussed in the following sections.

2.1. Batch-mode remote laboratories

Batch-mode remote laboratories are based on the principle of batch processing[20] that is the sequential execution of a series of programs ('jobs') on a computer without manual intervention. In the field of remote laboratories, the experiment requirements are submitted by the users in the form of sets of input parameters. These parameters prescribe the specific configuration of the experimental apparatus such that experiments can be run to completion without any manual intervention. Since this represents an automatic process, all input parameters are preselected through scripts or command-line parameters, which guarantee the feasibility and accuracy of the desired experiments with a specific set of input data (see Figure 1).

In general, batch processing has the following benefits:

- It allows sharing of computer resources among many users and programs.
- It shifts the time of job execution to when the computing resources are available.
- By keeping a high overall rate of utilization, the hardware cost is better amortized.

As is further detailed below, the batch-mode setup is adopted here as an example. The detailed structure of batch-mode remote laboratories is discussed in the following sections. Note that in batch mode, the response time is sacrificed which triggered the development of real-time architectures. Furthermore, real-time remote laboratories are often desirable because of their inherent user interactivity.

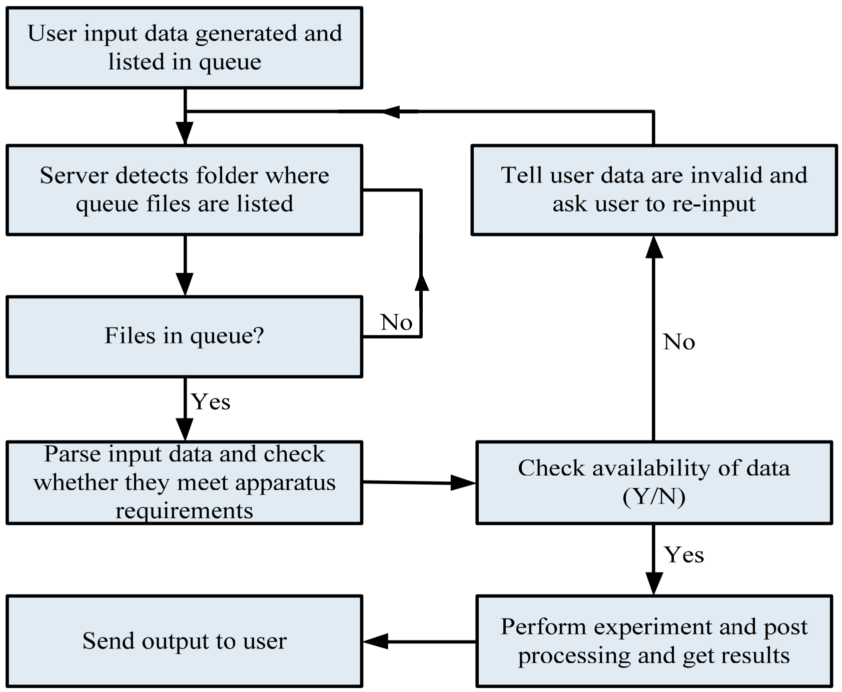

Figure 1: Basic procedure of batch-mode remote laboratory

Batch-mode remote laboratory architecture at SIT

A typical batch-mode approach was implemented using a client-server network that allows the concurrent execution of multiple experimental procedures. The core of developing a batch-mode system is to provide a server-side system that can communicate with the laboratory instruments so that an experiment can be carried out from the server side. In general, the batch-mode system developed at SIT is composed of several components: an internal distributed system that includes a server machine linked to the Internet, an internal controller PC linked to the server only, and an application system that is controlled through the PC with a data acquisition (DAQ) interface card. LabVIEW from National Instruments was used as the controller interface software. A chain client-server structure enables each component in the system to perform tasks individually, which provides great flexibility for different applications. The overall structure of the batch-mode remote laboratory architecture is illustrated in Figure 2.

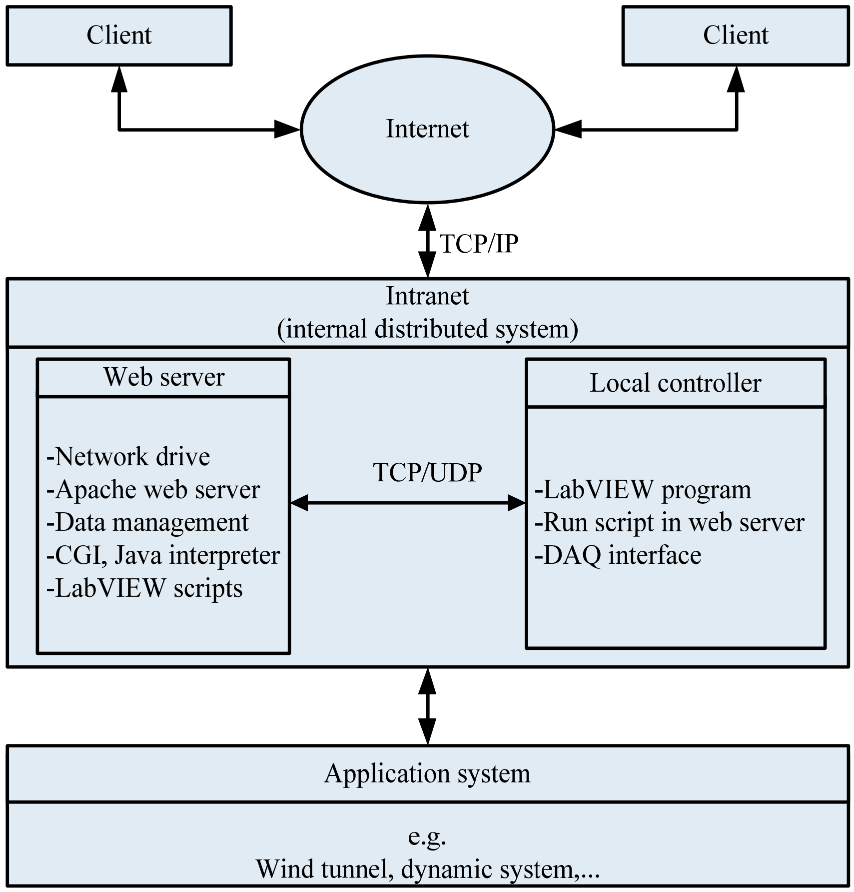

Figure 2: Batch-mode remote laboratory architecture

According to the batch-mode flow chart (see Figure 2), the procedure of employing this system for laboratory procedures is as follows. The clients provide the input parameters via a graphical user interface, which represents the general webpage (implemented in Hypertext Markup Language, HTML). Then, an input file is generated by a module programmed in Common Gateway Interface (CGI) in Perl language in the Web server and entered into an in-queue directory. The LabVIEW server program (located on the Web server and run by the local controller PC) continually monitors the user requests. If requests are found, the input file is parsed into variables that control the execution of the experimental procedure. Furthermore, the LabVIEW program generates an output webpage, which is presented to the user in the graphical user interface after a valid request has been generated. In the output webpage, the data are represented by arrays and plots using Java applets.

2.2. Real-time remote laboratories

In batch-mode remote laboratories, only one set of input parameters can be configured at once. As a result, the response time may become relatively long if several sets of input data are provided simultaneously by multiple users. For a particular user, real-time setups are desirable since they allow one to continuously and interactively modify input parameters and retrieve the corresponding output. In the context of real-time remote laboratories, the Graphical User

Interface (GUI) is of critical importance. Through the interaction with remotely accessible real equipment via the GUI, the users should be able to visualize the experimental process, gain a feeling of immersion into a real laboratory environment, and also be able to adjust the input and immediately observe the experimental output. Usually, the GUI is composed of an instrument control section, an experimental input section and an experimental results section. In the instrument control section, the options may include lighting, audio and video and data acquisition functions. In most real-time setups, a global video view providing an overview and a local video view zooming in on the analyzed object(s) are streamed in real time. In addition, the users are often given the option to save the video streams into a file.

Real-time remote laboratory architecture at SIT

Although real-time laboratories can be developed with different user interfaces (e.g. using LabVIEW), they can also be implemented using a client-server network approach that represents a three-tier Web architecture[21] as is illustrated in Figure 3. The first tier is the client, i.e. the user's client PC with Internet connection. The user interacts with the experiment through a Web browser that can remotely access the experimental setup. The middle tier is the resource manager, which consists of three servers to provide shared common services. The Web server is responsible for accepting and responding to the HTTP requests from clients and for authenticating authorized users. The database server is used for storing user records, experiment descriptions and experimental results. The schedule server is used to generate experiment schedules and to coordinate reservations for the experimental stations, thus preventing conflicts and congestions. The experimental setups represent the third tier of this architecture. The setups comprise the actual instrument(s), an instrument controller and a Web camera. The interface between the instrument controller and the experimental setup is realized with a DAQ card, which can be installed directly into the expansion slot of the computer acting as the instrument controller. The Web camera, which is employed for live streaming (and possibly recording) of the experimental procedure in the actual laboratory housing the apparatus, is connected to the Web server.

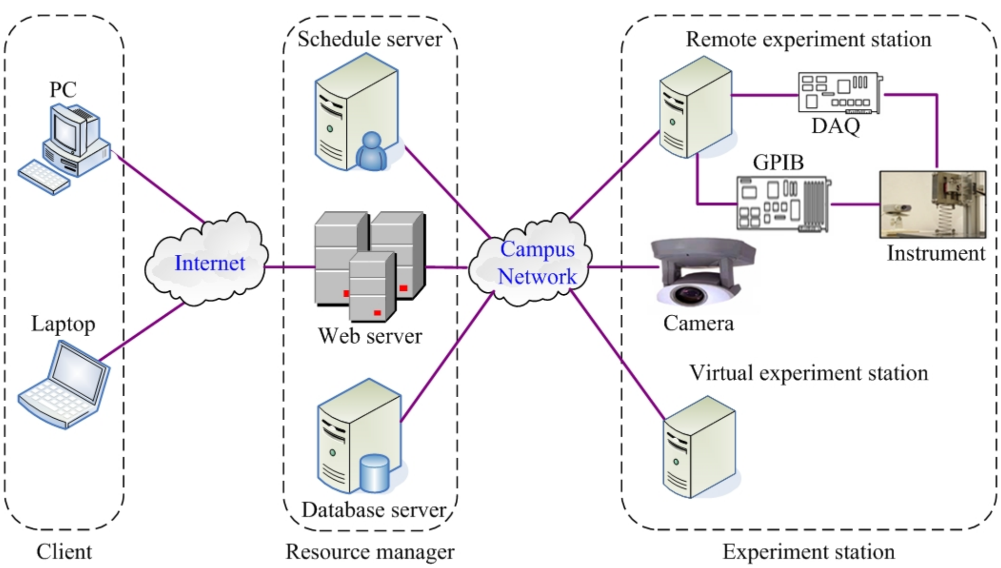

Figure 3: Real-time remote laboratory architecture [22]

3. A modularized architecture for remote-access laboratories

3.1. Software structure and modularization

The software structure discussed below is centered around a Web server and a local controller. Modularization can be accomplished based on the data flow chart shown in Figure 4in accordance with the software structure, both for batch mode or real-time mode. Although different experimental setups require different experimental procedures, batch or real-time, there can be generalized modules and logical relations that are independent of the specific application system in both modes. Therefore, separating modules that are responsible for certain specific functions can greatly reduce the work for remote laboratory developers, since most of the needed modules can be derived from these generalized ones by modifying certain parameters for the specific circumstances. Based on the discussion of batch-mode modules above, modules for certain responsibilities were separated according to the following flow chart (seeFigure 4).

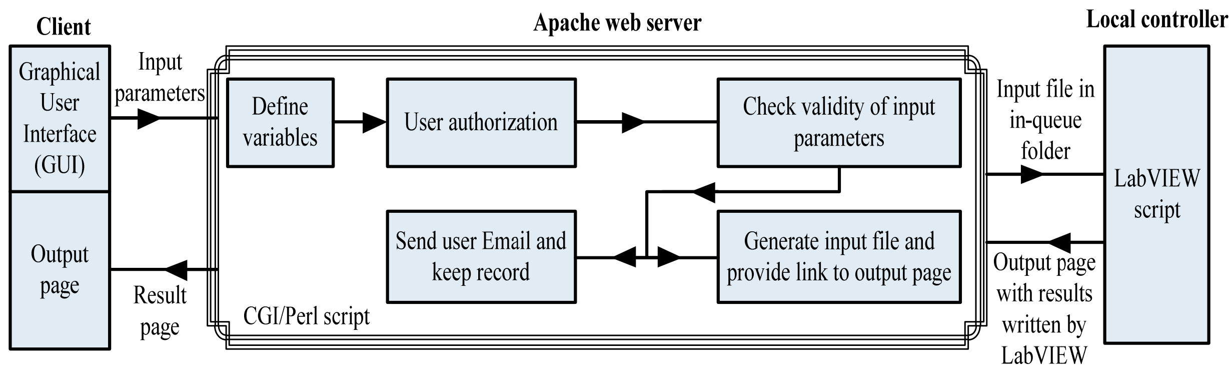

Figure 4: Software architecture of batch-mode remote laboratory

For the Web server, the Linux operating system was adopted since its highly stable performance greatly enhances the stability of the overall system. After the users submit their personal information and input parameters via the GUI, the server defines variables for all input data, checks whether the values of the input variables are in the allowable range, informs the user about the in-queue status, provides a link to the result page and generates an input file in a fixed format for the LabVIEW server to parse. After an experiment has been completed, an output page with the experimental results is placed into the out-queue on the server. These functions are independent of the specific experiments and thus can be separated into modules for a general batch-mode system. The modules here were implemented as scripts in Perl language. The communications beyond the border of the central box in Figure 4 are carried out through the in-queue and out-queue folders mentioned above. The structure and modularization inside the right box in Figure 4 is explained in the following section. The software architecture of the real-time setup is shown in Figure 5.

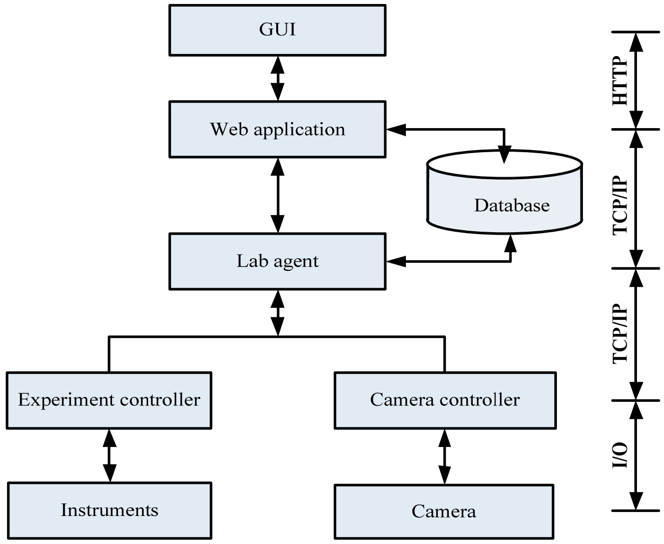

Figure 5: Software architecture of real-time remote laboratory [22]

The GUI that appears in the first layer presents the user with the available information and actions. Thus, it should be accessible from all platforms that are able to process HTTP, which ensures that it can be modularized for common use in the development of the real-time remote laboratory. The Web application layer processes the requests from the GUI and posts back the results for these requests. Among several dynamic Web content technologies for server side applications such as Hypertext Preprocessor (PHP), Java Script Pages (JSP), Active Server Pages (ASP), ASP.NET is most commonly adopted because it simplifies the developer's transition from Windows application development to Web development by offering the ability to build webpages containing controls similar to those of a Windows user interface. The lab agent, which is in the third layer, locates the experiment, routes the user request, returns the experimental results and stores them. Web applications and lab agents are all independent of the details of experiment implementation. Furthermore, since experiment controllers do not depend on the details of the database operation, the database can be separated. With these procedures, the reusability of the software and the efficiency of the development are attained.

3.2. LabVIEW server-side and modularization

The primary reason for employing LabVIEW to remote laboratory development is its ability to integrate with hardware to acquire data from real physical devices. LabVIEW has built-in engineering libraries which include in-line and off-line analysis and control. Hundreds of analysis functions in areas including signal processing, filter design, numerical computations as well as PID, vision and motion control are available in LabVIEW. Using these libraries in the same application where the measurement data are acquired simplifies the application development. LabVIEW is a graphical programming language. Using function blocks, wires and loops in place of text strings, one can create programs that look similar to the application flowcharts instead of translating that high-level design to specific text strings, thus avoiding errors in the translation from algorithm to code. Like text-based languages, LabVIEW compiles to machine code when running and performs at similar speeds to applications written in text-based languages. In addition to including its own optimized compiler at run time, LabVIEW continually compiles the programs during the design to help the developers catch errors while they are coding. Since LabVIEW offers powerful solutions for data acquisition from various hardware applications and features that enable data processing, it was employed for implementing the remote laboratory system at SIT. The procedures differ for batch mode and real-time mode, but in both cases, the general LabVIEW structure is illustrated in Figure 6.

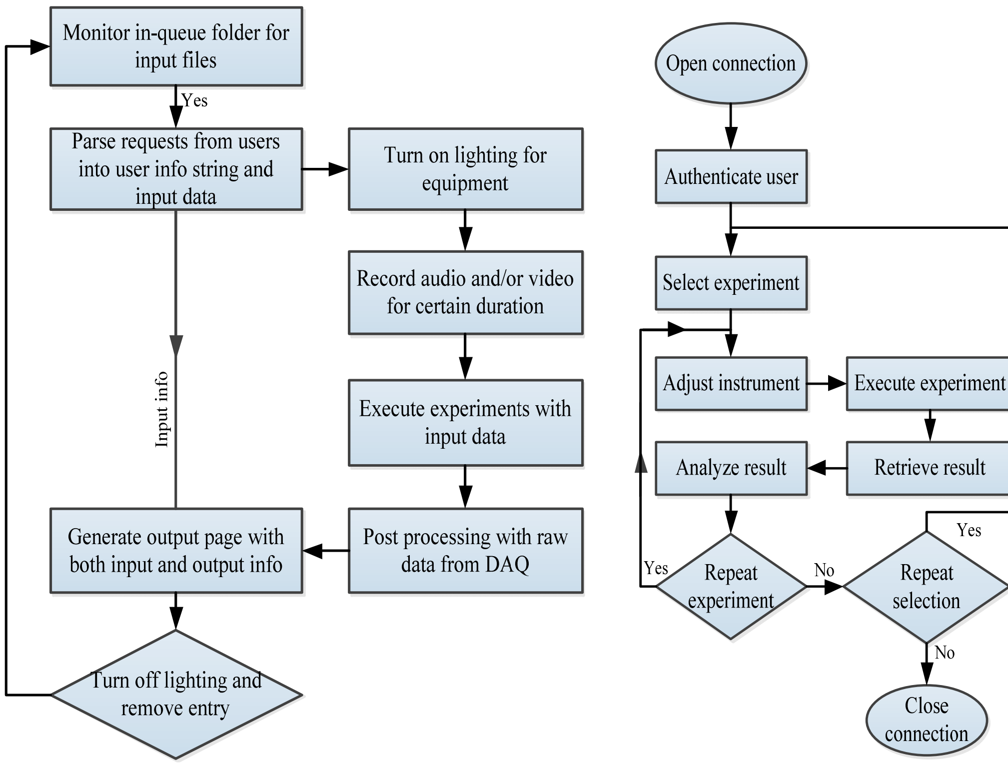

Figure 6: LabVIEW structure for batch-mode (left) and real-time (right) experimental setups

Modularization in developing remote laboratories is enabled also based on the structure of the LabVIEW script run by the local controller since one of the features of LabVIEW is the hierarchical nature of its Virtual Instruments (VI). After a VI has been created, it can also be used as a subVI in the block diagram of a higher-level VI. Therefore, a subVI is analogous to a subroutine in the traditional programming languages. Just as there is no limit to the number of subroutines one can use in a traditional program, there is also no limit to the number of subVIs one can use in a LabVIEW program. SubVI can also be nested, i.e. a subVI can be called from inside another subVI. When creating a LabVIEW application, one starts at the top-level VI and defines the inputs and outputs for the application. Then, one constructs the subVIs to perform the necessary operations on the data as they flow through the block diagram. If a block diagram contains a large number of icons, one should group them into a lower-level VI to maintain the simplicity of the block diagram. This modular approach makes applications easy to debug, understand and maintain.

As is demonstrated in Figure 6, certain subVIs can be developed based on the logical relation of functional modules. The sequence of the data flow and execution of different subVIs can be controlled by LabVIEW structural devices and timing functions. Also, most of the LabVIEW modules can be reused in the development of other remote experiments by simply employing these generalized VIs with their variables configured appropriately.

4. Implementation of a wind tunnel experiment using the modularization approach

4.1. Experiment description

Figure 7 shows the laboratory-scale FLOTEK 1440 wind tunnel housed in the undergraduate fluid mechanics laboratory at SIT with a golf ball inserted as the object to be investigated. This is a 'one-through' suction-type wind tunnel with a 12" × 12" test section. In the traditional hands-on experiments using this experimental setup, the students first turn on the power supply of the wind tunnel. Then, they adjust the air flow velocity in accordance with the requirements of the experimental procedure by selecting the appropriate rotational speed of the fan via a control panel. Finally, they measure the pressure distribution on the surface of the test object located in the test section using the manometer attached to the wind tunnel and calculate drag forces and coefficients from these experimental results.

Figure 7: Real wind tunnel

The golf ball experiment demonstrates the characteristics of flows over blunt (non-streamlined) bodies. A body immersed in a flowing fluid is subjected to various forces, including lift and drag. If the geometry of the body is symmetric about the horizontal and vertical planes and if the flow is parallel to both planes of symmetry, then there is only a drag force acting on the body. Because of flow separation, the theory describing drag is restricted to simple geometries such as a flat plate. Boundary layer theory can predict the separation point but cannot accurately describe the pressure distribution in the separated region. The effect of separated flow and the subsequent failure of boundary layer theory are often measured by the pressure distribution on a model using a dimensionless pressure coefficient C_p defined as:

$$C_p = \frac{P_s - P_\infty}{\frac{1}{2}\left(\rho / g_0\right)V_0^2} = \frac{P_s - P_\infty}{q}$$

Here, P_s is the pressure on the spherical surface, P_∞ is the pressure at infinity, ρ is the density of the fluid, g_0 is the gravitational constant, V_0 is the air flow velocity of the wind tunnel and q is the dynamic pressure. The difference between the high pressure in the front stagnation region and the low pressure in the rear separated region causes a large drag contribution called pressure drag. This is added to the integrated shear stress or friction drag of the body, which it often exceeds. The relative contributions of the friction drag and the pressure drag depend on the body's shape, especially its thickness. The effect of drag is often expressed by a dimensionless drag coefficient C_D defined as:

$$C_D = \frac{Drag}{\frac{1}{2}\rho V^2 A}$$

where *Drag* is the drag force measured in the experiment, ρ is the density of the air, V is the velocity of the free stream and A is the characteristic area.

4.2. Local control of experimental apparatus

In order to accomplish the goal of remotely controlling this physical apparatus via the Internet, the wind tunnel needed to be retrofitted with an electronic control system, which is capable of turning on and off the power supply, adjusting the air flow velocity by setting the fan speed and acquiring the resulting experimental data. Figure 8 shows the data acquisition module, which enables the sampling of 16 channels of pressure data at various conditions, the stepper motor driver and the pressure transducers, which allow for the connection of the pressure hoses directly to the DAQ board.

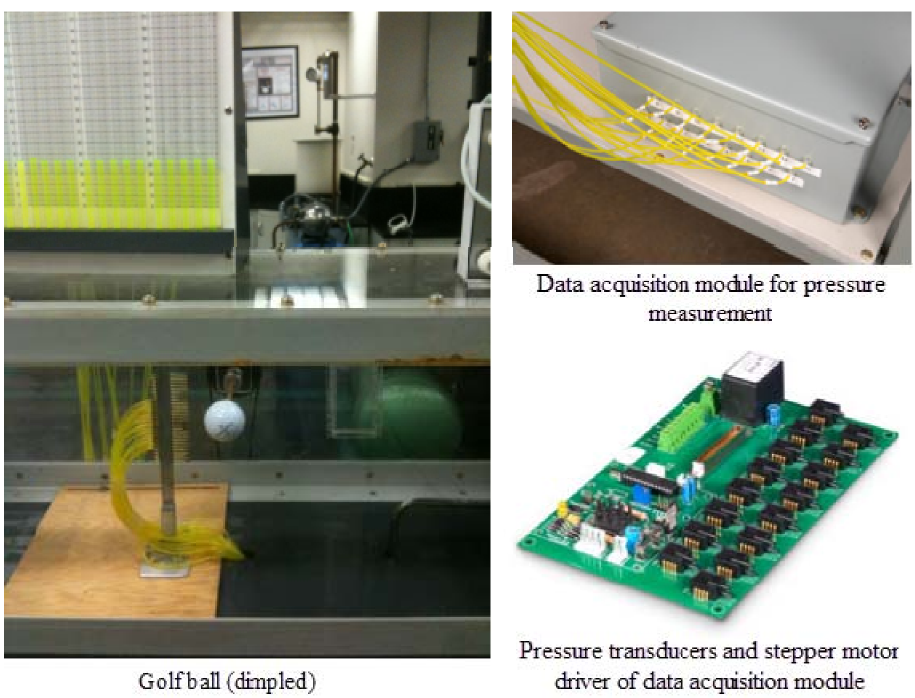

Figure 8: Wind tunnel with dimpled golf ball setup

Using this remotely controllable experiment, the students analyze several sets of air velocities. Given the high student enrollment of this course, a batch mode solution that is capable of dealing with large number of potential users was selected. In this system, the fan motor can be activated and its speed adjusted remotely, and after the completion of an experiment, the pressure distribution is provided to the user. Figure 9 depicts the schematic of the wind tunnel control system at SIT.[23]

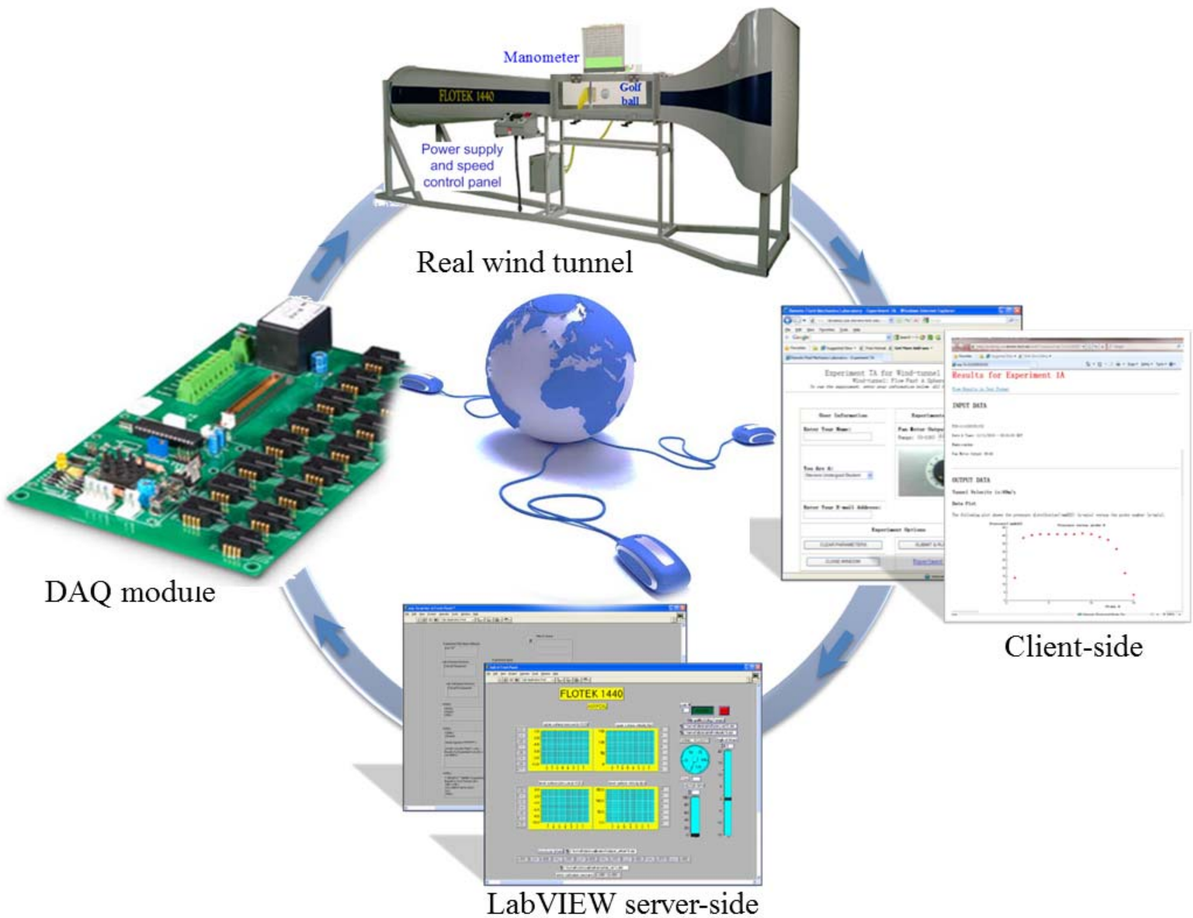

Figure 9: Schematic of remotely operated wind tunnel control system

As is shown in Figure 10, the client-side provides the user with a means for sending commands from the Web server to the application server and allows the experimental configuration and parameters to be controlled. The Web server is a PC that bridges the client Web browser to the Web server application. The interface between the server and the experimental setup is realized with the DAQ card, which can be installed directly in one of the expansion slots of the server. The DAQ card performs the analog input/output functions and allows the software to communicate with the sensors and actuators of the experimental setup using low-level analog and digital signals. The Web server is connected to a servo-camera, which is used to stream and record video from the experiment.

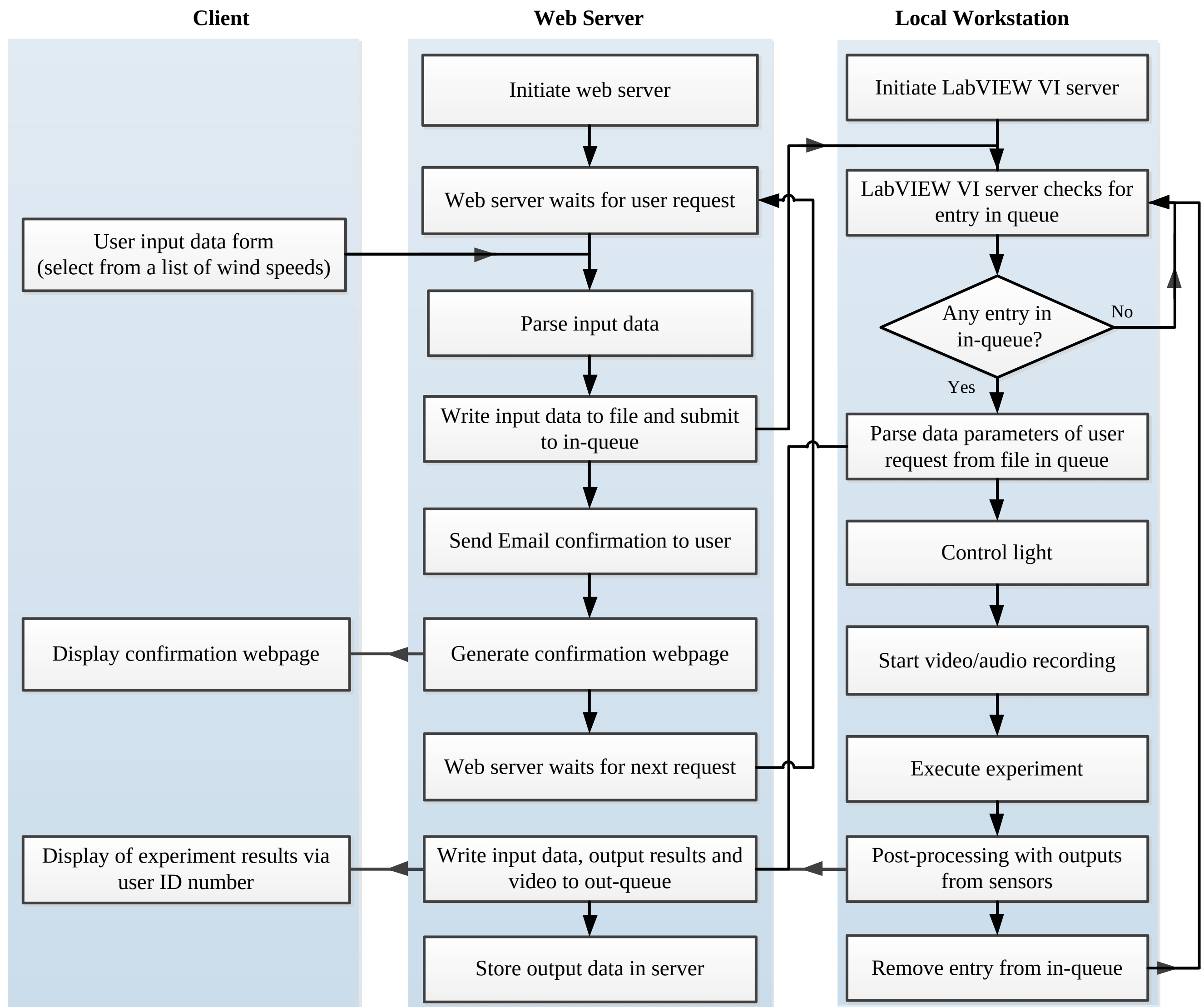

Figure 10: Data flow structure for remote-access wind tunnel laboratory

5. Conclusions

This paper describes some recent developments that were accomplished as part of a multi-disciplinary research project on online laboratories at SIT with funding from the National Science Foundation. The concept of modular programming was applied to develop a modularized and scalable system architecture for remote experimentation, which enables the interaction of multiple users with a network of spatially distributed experimental devices. This approach facilitates the standardized description of remote-access laboratories so that other experiments can be developed more easily and quickly. Taking advantage of the modularization approach, a remote-access wind tunnel experiment was implemented.

Acknowledgment

This work was supported by NSF Grant No. 0326309. This support is gratefully acknowledged.

References

[1] Ozbek, M. E., Kara, A. & Atas, M. (2010). Software technologies, architectures and interoperability in remote laboratories. *Proceedings of the 9th International Conference on Information Technology-based Higher Education and Training*, pp. 402-406.

[2] http://www.eng.fsu.edu/~dommelen/courses/cpm/notes/progreq/node2.html, accessed on 02/28/11.

[3] Azad, A. K. M. (2007). Delivering a remote laboratory course within an undergraduate program. *International Journal of Online Engineering*, Vol. 3, No. 4, pp. 27-33.

[4] The benefits of modular programming, available at: netbeans.org/project_downloads/usersguide/rc-book-ch2.pdf.

[5] Esche, S. K. & Chassapis, C. (1998). An Internet-based remote-access approach to undergraduate laboratory education. *Proceedings of the Fall Regional Conference of the Middle Atlantic Section of ASEE*, pp. 108-113.

[6] Esche, S. K., Tsatsanis, M. & Prasad, M. G. (1998). Development of a remotely accessible dynamical systems laboratory for undergraduate teaching. *NSF-ILI Award #9851039*.

[7] Esche, S. K., Chassapis, C., Nazalewicz, J. W. & Hromin, D. J. (2003). An architecture for multi-user remote laboratories. *World Transactions on Engineering and Technology Education*, Vol. 2, No. 1, pp. 7-11.

[8] Esche, S. K. (2005). On the integration of remote experimentation into undergraduate laboratories - pedagogical approach. *International Journal of Instructional Media*, Vol. 32, No. 4, pp. 397-407.

[9] Enloe, C., Pakula, W. A., Finney, G. A. & Haaland, R. K. (1999). Teleoperation in the undergraduate physics laboratory - teaching an old dog new tricks. *IEEE Transactions on Education*, Vol. 42, No. 3, pp. 174-179.

[10] Brofferio, S. C. (1998). A university distance lesson system: experiments, services, and future developments. *IEEE Transactions on Education*, Vol. 41, No. 1, pp. 17-24.

[11] Active Robotics, available at: http://www.robotics.reading.ac.uk/.

[12] Esche, S. K., Prasad, M. G. & Chassapis, C. (2000). Remotely accessible laboratory approach to undergraduate education. *Proceedings of the 2000 ASEE Annual Conference & Exposition*, Session 3220.

[13] Esche, S. K. (2006). On the integration of remote experimentation into undergraduate laboratories - technical implementation. *International Journal of Instructional Media*, Vol. 33, No. 1, pp. 43-53.

[14] Bhandari, A. & Shor, M. H. (1998). Access to an instructional control laboratory experiment through the World Wide Web. *Proceedings of the American Control Conference*, pp. 1319-1325.

[15] Ko, C. C., Chen, B. M., Chen, S. H., Ramakrishnan, V., Chen, R., Hu, S. Y. & Zhuang, Y. (2000). A large-scale web-based virtual oscilloscope laboratory experiment. *IEEE Engineering Science and Education Journal*, Vol. 9, No. 2, pp. 69-76.

[16] Ko, C. C., Chen, B. M., Chen, J., Hu, S. Y., Zhuang, Y. & Tan, K. C. (2001). Development of a web-based laboratory for control experiments on a coupled tank apparatus. *IEEE Transactions on Education*, Vol. 44, No. 1, pp. 76-86.

[17] Shaheen, M., Loparo, K. A. & Buchner, M. R. (1998). Remote laboratory experimentation. *Proceedings of the 1998 American Control Conference*, pp. 1326–1329.

[18] Shen, H., Xu, Z., Dalager, B., Kristiansen, V., Strom, O., Shur, M. S., Fjeldly, T. A., Lu, J. Q. & Ytterdal, T. (1999). Conducting laboratory experiments over the Internet. *IEEE Transactions on Education*, Vol. 42, No. 3, pp. 180-185.

[19] Li, Y., Esche, S. K. & Chassapis, C. (2006). A Web services approach for sharing remote laboratory resources. *Proceedings of the ASME International Mechanical Engineering Congress and Exposition*.

[20] http://en.wikipedia.org/wiki/Batch_processing, accessed on 02/28/11.

[21] Li, Y., Esche, S. K. & Chassapis, C. (2007). A framework for sharing online laboratory resources. *Proceedings of the ASME International Mechanical Engineering Congress and Exposition*.

[22] Li, Y., Esche, S. K. & Chassapis, C. (2008). A scheduling system for shared online laboratory resources. *Proceedings of the 38th ASEE/IEEE Frontiers in Education Conference*.

[23] Aziz, E.-S., Esche, S. K., Chassapis, C., Dai, S., Xu, S. & Jia, R. (2008). Online wind tunnel laboratory. *Proceedings of the ASME International Mechanical Engineering Congress and Exposition*.