2006-101: WEBLAB – COMPREHENSIVE REMOTE LABORATORY SYSTEM

Zachery Allen, Bismarck State College
Mr. Allen has over 9 years of technical experience in the Computer Industry, the last six of which have been specifically in the Electrical Industry. He presently works for Bismarck State College (BSC), leading development of new and interactive tools for teaching online. He previously led the development of a degree program for Electrical Transmission Systems Technology (ETST). In addition he is the resident expert on transmission systems operation and helps manage relations with the Electrical Industry.

Daniel Schmidt, Bismarck State College
Mr. Schmidt has over 23 years of experience in the Process Industry, with the last six spent in Energy Education. Mr. Schmidt currently works for Bismarck State College (BSC) as the program manager for BSC's Process Technology and Electrical Transmission Systems Technology programs. He previously instructed and developed courses for BSC's Energy programs, and is an expert in process technology.

William Wahlman, Bismarck State College
William Wahlman is Director of Energy Technology Online Programs at Bismarck State College, Bismarck, N.D. He worked eight years as a power plant operator and trainer before joining BSC in 1996 through a partnership with a northern Minnesota plant. For two years, he conducted on-site technology courses that enabled 33 workers to earn AAS degrees and certificates. An extensive outreach developed and Wahlman converted BSC's 20 power plant technology courses for online instruction in 1998. He helped develop two additional online programs - Electric Power Technology and Electrical Transmission Systems Technology - in cooperation with industry.
WebLab – Comprehensive Remote Laboratory System

WebLab is the name for a remote laboratory system used at the technical education level by Bismarck State College to provide hands-on laboratory experiences to online and place bound students. The WebLab system is unique in that it not only provides access to remote equipment, but is an educational system with aspects such as instructional design based objectives, pre- and post-assessment, scheduling, and lecturing built into the system at the core level. Furthermore, the equipment access portion of the system is built around industry standard instrumentation and process control equipment and communication. This allows laboratories to be built and controlled using off-the-shelf equipment from multiple vendors. While educational use of supervisory control and data acquisition (SCADA) systems is relatively new, the technology has been used for decades in the Energy, Automotive, and Food Processing Industries to name a few. By using industrial components, the cost to develop laboratories has been reduced while at the same time increasing reliability and availability. Lastly, the WebLab system is designed to be used over the public Internet and is optimized for use over data connections as low as 56kbps. The user interface requires no software to be installed on student PCs, will work through corporate firewalls, and can refresh as many as 100 data points within a second’s time.

The Purpose of WebLab:
The WebLab system was originally designed to allow remote use of equipment to students via the public Internet. During its four years of development, however, it has evolved into much more. Providing remote access was just the start, and while the WebLab system itself is unique, it is just a piece of a larger project. One of the main focuses of the WebLab project, which is funded by a National Science Foundation Course Curriculum Laboratory Improvement (CCLI) grant (Award #DUE-0340927), is to help establish remote laboratories as a successful academic tool. This challenge is two fold. The first hurdle is to validate that student learning can occur through remote means. The second hurdle is to inform the rest of the academic community that remote laboratories are viable, cost-effective, and genuinely promote student learning.

The first hurdle can be addressed through high quality design, not only of the system and laboratory, but also through the use of a comprehensive system with good instructional design. This not only will help validate results, but will help students as well. For example, students who are new to an industry or set of objectives may become overwhelmed if exposed to a full laboratory with no preparation. That is why the instructional portion of WebLab is designed to slowly build students up to using the full system. Pre-tests, animated lectures, assignments, reading, and quizzes may all be employed before a student gains access to the equipment for the laboratory exercise. During the exercise, laboratory data can be logged and then trended and analyzed by the system and instructors. Lastly, post-exercise testing and additional learning materials can be used to help solidify learning outcomes.

The second hurdle can be surmounted by making remote laboratories serve as learning platforms instead of simply remote equipment access platforms. Any electrical utility’s instrumentation and control technicians can explain how remote access and control works. The true challenge is in creating a learning environment. It is up to us to raise the bar and show the rest of the academic community that learning is possible and efficient through the use of remote laboratory systems.
The goal of this paper is to share some of the problems, challenges, successes, and mistakes that were made during the development of the WebLab project. From how equipment and labs can be used to how and why this project evolved from a SCADA system to a robust educational system, this paper shall allow others to bring remote laboratories into their institutions with shorter development times and higher rates of success while disseminating information that promotes the use and validity of remote laboratories.

System Overview:
Before delving into the challenges and successes of WebLab, a brief overview of the system and its architecture is in order. The WebLab system is composed of three main subsystems as shown in Figure 1 and is designed to continue to run in the event of a failure of any one subsystem.

The first subsystem is the physical laboratory itself including all the transducers, controllers, actuators, and programmable logic controllers (PLCs) needed to perform the laboratory activities. The laboratory currently being built is a miniature power system called the Power Grid Laboratory (PGL) and consists of 2 three-phase motor generator sets, three substations, three transmission lines, and over 100 data points, 40 of which are analog. Figure 2 illustrates the one line diagram of the PGL.
The second subsystem is the database and web applications, including the real-time laboratory database, student database, instructional database, scheduling system, and the client access system. The databases are Structured Query Language (SQL) compliant and are compatible with multiple Relational Database Management Systems (RDBMS). The current version runs on Microsoft SQL Server 2000.

The third major subsystem is the user interface. Much effort has been made to design the user interface as a replica of the industry specific equipment, controls, and displays that students would find in the real world. Additionally, the interface must be used for all aspects of the system by both students, for doing laboratory work, and instructors for grading, monitoring, and facilitating the laboratory exercises.

When all three subsystems are combined, the complete system allows robust access and learning opportunity, with high speed and reliability, all with an easy to use interface.

The topics discussed in this document focus on general lessons learned from the WebLab project. For specific details on the hardware implemented for all three major versions of the WebLab system, please reference Appendix A.

**General Control System Considerations:**
Control and data acquisition (referred to here as control) is the most critical aspect of developing remote access to or automation of laboratory equipment. Without a solid and reliable control system, the rest of the system cannot function nor can learning occur. Of the all the areas of
design and build of the WebLab system, control has caused more issues and been the source of more redevelopment than any other.

Control of a remote laboratory must be addressed in terms of both its overall control system needs and specific equipment control needs. Overall control system considerations deal with high-level laboratory wide concerns such as control speed, logic, resolution, etc. Specific equipment control considerations are concerned with low-level data point or equipment details such as transducers, sensors, motor controllers, etc. Each is a topic in its own right and while overlapping at times will be discussed separately.

A good place to start when designing a remote laboratory is with the overall control system requirements. This includes such things as the type of data points needed for the laboratory to meet its objectives. Data points can consist of data inputs (system telemetry), data outputs (system control) or both. Additionally, data inputs and/or outputs (I/O) need to be defined as either digital (on/off) or analog (variable). It should be noted that analog I/O points are significantly more expensive and complicated to use than digital I/O points. Two major considerations when selecting analog I/O points are their resolution and acquisition speed. Resolution refers to the accuracy at which the analog values are converted to digital values and acquisition speed refers to the number of samples per second the I/O is capable of. Higher resolutions and speed can add cost and should be determined by the specific needs of a laboratory. For example, if the laboratory interface is used over the public Internet, it may not be effective for ten thousand samples per second to be streamed to the user; therefore a lower speed acquisition solution could be implemented. In the case of the WebLab PGL, the laboratory uses both high speed sampling for local and special operations and low speed sampling for updating user data. It is important to design for maximum flexibility but also to be realistic about what speed is actually needed to effectively operate the laboratory.

**Control System Types:**

Once general requirements such as I/O type, number of I/O points, resolution, and speed have been determined, the next major general consideration is the interface and logic, i.e. how will the control hardware be connected to the control logic and other systems, as well as what type of control is actually needed. When determining the type of controller and logic needed for a remote laboratory the overall control requirements should be defined. This is dictated by the operation of the laboratory and includes such aspects as simple data sampling, simple control, database interaction, and automation or advanced control requirements, such as feedback control loops using Proportional Integral Derivative (PID) algorithms. For example, the first version of WebLab used only digital I/O and had very simple control and data sampling with only a few database read/write operations. It was very easy to write control logic software for this version. In contrast, the current version has six PID feedback control loops, one primary control and data sampling control loop, one database communication control loop, one safety control loop, and one maintenance control loop. Furthermore, all of these control loops run simultaneously and independently. Knowing what type of logic and control are needed and how they are needed to run (linear, simultaneously, etc.) will help determine which type of control hardware and software is best suited for a remote laboratory.
The interface and control logic of the WebLab system has gone through three generations. The first generation used serial control devices with digital I/O controlled by a Microsoft Visual Basic (VB) application. The second generation used PCI based control devices with digital and analog I/O controlled by a VB Microsoft Windows service. The current generation uses PLCs controlled by firmware loaded logic. Each of these types of controllers has advantages and disadvantages as illustrated in Table 1.

<table>
<thead>
<tr>
<th>Control System</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
</table>
| Serial I/O     | • Ease of Use  
• Inexpensive  
• Easy Software Integration | • Non-Synchronous  
• Slow Sampling Speed  
• Difficult to Program PID Controls  
• Requires Separate Transducers |
| PCI DAQs       | • Large Variety of Data Points  
• High Speed  
• Standard Programming APIs | • Require Dedicated PC with Proper Interface  
• Logic run on host PC  
• Difficult to Program Multiple Simultaneous PID Controls  
• Requires Separate Transducers |
| PLCs           | • Large Variety of Data Points  
• Integration with other control devices (i.e. Serial I/O, TCP/IP, OPC)  
• High Speed  
• Logic run on PLC  
• Multiple Simultaneous Control Loops & PID Controls  
• Built-in PID Algorithms  
• Industry Standard  
• Transducer Built-in I/O Cards | • High Cost  
• Proprietary Programming Software & Languages |

Table 1

The reason the WebLab system was redeveloped to use PLCs was the laboratory’s requirement for reliable multiple simultaneous feedback control loops. The PLCs used are capable of simultaneously running 16 independent PID control loops while running six standard control loops. Additionally, some interesting benefits occurred by implementing PLC control. These included reduced wiring and labor due to PLC cards having integrated transducers, easier system integration due to the serial and TCP/IP support of the PLCs, and lower cost upgrades as I/O cards can be added or removed with incremental cost.

Low-Level I/O Considerations:

With the laboratory’s general considerations determined and a viable control system in place, the next step in developing a remote laboratory is determining how to control and receive data from the actual laboratory equipment. This is a major part of the development as laboratory equipment and control systems do not always speak the same languages. Control systems are composed of precise pieces of equipment that operate like computers; therefore, data from laboratory equipment has to be adapted to the data that a precision controller might need. Fortunately there are some standards for both digital and analog I/O points.

Most control system equipment requires digital I/O points to use TTL logic, i.e. 0-5 volts DC (low/high) with very low current. Each digital input point in a laboratory needs to be able to
have its native signal (a switch setting, voltage presence, current presence, flow presence, etc.) converted to TTL logic. TTL logic can easily be converted for laboratory use by using Darlington Transistors or Solid State Relays (SSRs). The higher voltage and current switched through the Darlington or SSR can then be used to operate equipment such as on/off valves, relays, mechanical contactors, lights, etc.

Analog I/O use standard DC signals in the ranges of 0 to 5V, 0 to 10V, -5 to 5V, -10 to 10V, 0 to 20ma, 4 to 20ma, 0 to 1ma, or -1 to 1ma. Current based systems are often used in industrial automation because they are not affected by signal deviations caused by resistance in long cable runs. Choosing the right signal can be frustrating and usually depends on what signals the control hardware and transducers support. Some can be configured dynamically; however, most are set from the manufacture.

Transducers and controllers are needed to convert the low voltage/current DC control signals for use in a laboratory. Many types of controllers and transducers can be built or purchased to meet a laboratory’s needs. Pulse Width Modulation (PWMs) controllers and Variable Frequency Drives (VFDs) can be used to control DC and AC motors respectively. Servos and stepping motors can be used for positioning; linear actuators, flow control valves, and just about anything else needed to meet a laboratory’s objectives most likely have a controller that is compatible with standard control system analog I/O signals.

The original version of WebLab had all home-built transducers. As the project progressed and the transducers became more numerous and complicated (such as those measuring reactive power and phase angle between two generators), a decision was made to eliminate all custom built transducers. This was done for two reasons. The first was reliability. While all the transducers had worked well, they were custom. Even with good documentation, if one failed and the designer or electronics professionals were not available, extended down-time could ensue. Every transducer and controller in the WebLab PGL today is replaceable with an off-the-shelf component, some of which are kept in stock with the laboratory. The second reason was cost. Where applicable the transducers are built into the PLC input cards saving time, wiring, and money.

Finding the right transducer or controller can be quite difficult as most instrumentation and control vendors provide customers with a plethora of ordering options. Deciphering the ordering options and getting quotes on only a few transducers can be hectic as most of the vendors are used to dealing with large industrial projects and standard industrial equipment. However, most sales and engineering support people will work through your need with you. From usability, to maintenance, to cost, control system design is one of the most involved aspects of developing a remote laboratory. Be prepared to spend a significant portion of a remote laboratory project on these issues.

Lastly, when searching for transducers and controllers make sure to check all areas of industrial automation and control, especially those which bear no relation to the remote laboratory. For example, the PWM motor controllers that act as the generator governors in the PGL were designed to run conveyor systems at warehouses and were hundreds of dollars less than controllers made for the specific motors used in the laboratory.
Remote Control Integration:
With a viable control system in place, the next major challenge is how to integrate control of the laboratory into a system that can be used over the public Internet. The first question that needs to be addressed is what type of control schema to implement. The answer should be based on the design and requirements of the laboratory. The two major types of controls schemas discussed here are direct and indirect control. For the purpose of discussion, direct control means users can directly access the laboratory’s control hardware, and indirect control means there is some middleware between the user and the control hardware.

Each type of control has advantages and disadvantages based on how the control schema is implemented. Allowing direct control of equipment is easier and eliminates some of the architecture and systems needed for indirect control. Direct control is well suited to smaller laboratories that do not have major safety concerns and do not require simultaneous access by multiple users. An example of direct control would be the original version of WebLab where users could set values on and receive data directly from the serial controllers through the user interface. This caused some issues that resulted in the migration of the laboratory to indirect control. One of the major issues was that if a user was accessing the controller then any automated control logic could not have access until the user request was complete. Another issue was that two users could not request data or issue commands at the same time. Due to these issues, an indirect control schema was developed where system data and control requests are stored in a real-time laboratory database.

Indirect control for the WebLab system is accomplished by creating a central clearinghouse for all laboratory data. Users read data from and issue desired commands to the database. Since the database uses a SQL compliant Relational Database Management System (RDBMS) it can handle multiple user requests without issue, and it can also queue and prioritize them. In this way multiple users can view laboratory telemetry from multiple locations without impacting the actual control hardware or creating a programming headache. While there is only one laboratory, many students and instructors can view the laboratory together. Students could be assigned to control different parts of the laboratory and have to work together while an instructor watches the system as a whole or even initiates faults on the system. By letting the database handle this work load, the system is free to operate efficiently and reliably.

When the indirect control schema was combined with PLC control hardware an unapparent advantage was realized. That advantage was safety. Safety is a major concern of the PGL since real electrical equipment using lethal amounts of electricity is used by the laboratory (DC voltages of up to 150V 5A, AC voltages up to 450V 2A, motors speeds up to 2500 RPM). Since the PLCs operate independently of the database, they view user commands as “requests.” If a request could or does result in a potentially unsafe condition the PLCs can ignore the command or rectify the problem automatically. Furthermore, the PLCs can sense the status of the user connection and database and if either is interrupted (e.g. Internet failure, server crash, etc.) the PLCs can safely shut down the laboratory without causing harm to personnel or equipment. The PLCs’ ability to run multiple control loops simultaneously allows these safety checks to take place independently of and faster than any user command that is issued. Additional PLC based
safety could be implemented, such as using motion detectors to determine if person is standing too close to the laboratory equipment or if someone enters the laboratory room.

**Database Considerations:**
Database design, operation, and loading have an impact on a remote laboratory’s performance. A number of lessons were learned on the WebLab project concerning system databases. One early problem with the WebLab system was the consolidation of the real-time laboratory database and user database. The problem with this configuration was transaction logging of the database. When a database is created, the backup and recovery options for the database need to be configured. Since losing student data is not an option, the WebLab database was configured to use full transaction logging and backups, meaning that all updates to the database were logged in a file and could be recovered if there was a failure; this is typical of most databases. When the laboratory was running at full capacity, the resultant transaction log files for updating over 100 data points twice a second were incredibly large. While this didn’t cause any devastating performance problems, it was wasteful and inefficient. The system was redesigned by splitting the database into two new databases one for user and academic information and one for the real-time laboratory data. The user and academic database uses a full database backup and recovery scheme, whereas the real-time database uses neither a transaction log nor data backup. Since the data in the real-time database is constantly refreshed and is only valid for the last sample, there is no reason to log or back it up. This design change significantly reduced processor and disk loading on the database server.

Another database consideration is the use of stored procedures. Stored procedures are prewritten bits of SQL code that are executed on the database server. By using stored procedures processing can be removed from the control program or PLCs and moved to the database server. In the case of WebLab, the PLCs were quite busy while the database consumed less than five percent of the server’s resources. By using stored procedures, thereby moving some of the processing to the database, PLC resources were freed while the database server was only nominally impacted.

Lastly, control system to database communication is an important consideration. The frequency and number of data points which are updated can affect the performance of a remote laboratory system. While laboratory control hardware may require very high sampling rates to operate, the database and user information may not need such high speeds. Furthermore, data may not need to be sent from the control system to the database if no user is accessing the laboratory. The WebLab system uses many resource saving options while communicating with the real-time database. First, data is only updated while users are accessing the laboratory. Second, only data points that have changed are updated. Values that have not changed are not sent to the database (this reporting technique is sometimes referred to as “reporting by exception”). Lastly, the PLCs update the database only twice a second even though they are acquiring data and running control loops as fast as once every 2 ms.

**Internet Considerations & User Interface:**
Even with a reliably operating control system and sound remote control schema, users still need to effectively interact with the remote laboratory system. Numerous considerations were made when selecting the WebLab’s user interface and even more considerations were made when
developing it to work on 56K modem connections over the public Internet. Fortunately, the selection of Macromedia Flash as the user interface platform and an indirect control schema resolved some of the Internet issues.

Of course the major Internet issue is bandwidth, but there are a number of others such as latency, dropped connections, firewalls, software installs, etc. The indirect control schema mentioned earlier resolved most problems with latency and dropped connections. The PLCs can accommodate latency issues by processing commands as requests at any incoming rate or time, as well as by separating telemetry requests from control requests (telemetry requests terminate at the database and do not reach the PLC). This is in part possible because the PLCs have direct control of the hardware and handle all time sensitive actions needed for the laboratory’s operation without requiring user input. The PLCs can also accommodate dropped connections by initiating a timeout period if a user connection is lost. If the connection is reestablished within the timeout period the session continues as normal; however, if the timeout is reached the system shuts down and reinitializes itself to be ready for the next user connection. Lastly, system utilization is conserved because the PLCs do not update the database if no user connections are established. Why update over 100 data points if nobody is viewing them?

Macromedia Flash as an interface platform helped to resolve a number of user issues, such as having to install software on user computers to allow access to the laboratory. This is due to the fact that the Flash plug-in is often bundled with web browsers such as Microsoft Internet Explorer. Macromedia Flash also helped to resolve firewall issues as Flash allows interface and system data to be received using TCP/IP port 80. Port 80 is the port designated for HTTP and if users are allowed to view web sites then port 80 must be open on their firewall therefore allowing access to the WebLab system. As a vector based, object orientated platform, Macromedia Flash helped keep the user interface quite small and efficient (currently 130k for the entire interface). Additionally, Flash files run using the local computer’s resources thereby allowing all interface processing to be handled locally by user hardware thus removing this processing load from the client access system and server.

Telemetry and control data also has to be addressed. Since the WebLab system updates over 100 user interface data points at least once a second, an efficient method needed to be developed to connect the user interface to the client access system and ultimately the laboratory database. The original version of the WebLab user interface exchanged data with client access system using XML. XML was easy and standard, but it was soon found that the additional data that composed the XML tags was much more than the actual data. Therefore the XML data formatting was abandoned for the much simpler HTTP data formatting. Since HTTP formatted data does not have the tags associated with XML, this change drastically reduced the amount of data sent between the user interface and client access system. Table 2 compares example code in both XML and HTTP formatting for five identical data points. In this example the XML formatted data is over 600 percent larger than the HTTP formatted data. While comparing data sizes this small seems trivial, it becomes a large issue if refresh rates are to be kept high over modem connections in rural areas of the country. Furthermore, the refresh rate of the system telemetry throttles based on the speed of the user connection. It will refresh as slow as once every five seconds and as fast as twice a second. Control requests are only sent upon their execution. The current data stream which updates all of the WebLab PGL’s telemetry is only 1,230 bytes. As of
now the system has been successfully tested on modem connections as slow as 24.6k connection rates.

<table>
<thead>
<tr>
<th>Format</th>
<th>XML</th>
<th>HTTP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example Data</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```xml
<?xml version='1.0' encoding='ISO-8859-1' ?>
<WebLab>
  <Analog_Points>
    <Point>
      <PointID>WebLab_M64_P00</PointID>
      <PointValue>0.01</PointValue>
    </Point>
    <Point>
      <PointID>WebLab_M64_P01</PointID>
      <PointValue>1.71</PointValue>
    </Point>
    <Point>
      <PointID>WebLab_M64_P02</PointID>
      <PointValue>0.08</PointValue>
    </Point>
    <Point>
      <PointID>WebLab_M64_P03</PointID>
      <PointValue>1002</PointValue>
    </Point>
    <Point>
      <PointID>WebLab_M64_P04</PointID>
      <PointValue>45.12</PointValue>
    </Point>
  </Analog_Points>
</WebLab>
```

Table 2

<table>
<thead>
<tr>
<th>Format</th>
<th>XML</th>
<th>HTTP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>716 Bytes</td>
<td>114 Bytes</td>
</tr>
</tbody>
</table>

Figure 3
Thus far the discussion of bandwidth has been concerned with the actual interface download and laboratory telemetry and control. Another feature of the system is live video; however, users wishing to use the live video features of the laboratory would need a high speed Internet connection. While having or not having the live video does not affect the operation of the laboratory it adds to the realism of the laboratory. Figure 3 is a screen capture example of an actual WebLab video feed.

The user interface has been designed to mimic real world controls and instrumentation as much as possible. Exposing students to the look and feel of the equipment that they will be operating in the real world can only help when they enter the workforce. This level of realism allows students to learn topics ranging from how to accurately read analog gauges to which way a pistol grip control lever is turned to trip or close a power circuit breaker. By building flexibility into the interface and client access system, new and old types of virtual controls can be used to act as laboratory controls. Figure 4 is an example of the Flash interface used on the development version of the WebLab PGL.

Conclusions & Future Work:
To this point the technical and design challenges and successes of the WebLab system have been presented. While the technical aspects of the system are a project of their own, a whole other avenue of development is needed to implement a successful remote laboratory. These are the
instructional and ancillary systems. The original plan for the WebLab system was to create a system that allowed remote access to laboratory equipment. That plan has come to fruition; however, it became apparent early in the project that much more is needed to make the system a viable educational platform.

The term instructional systems is inclusive of all the learning materials, lab manuals, lessons, tests, quizzes, assessment, and laboratory activities that go along with any given laboratory. The term ancillary systems is inclusive of systems designed and needed to support the operation of the laboratory such as the student database, scheduling system, data logging and trending, etc.

While the WebLab project succeeded technically, no time or resources were allotted for instructional and ancillary systems development (regardless, the developing has been and is being done). Fortunately, this shortcoming was detected early enough in the development that the system was redesigned to be able to integrate the instructional and ancillary systems as they are developed. This gives flexibility as additional instructional systems or lessons can be added to the system as the laboratory is upgraded or new topics are integrated into related curricula. A number of laboratory lessons to be integrated into the instructional system have been designed for the PGL and include pre-testing, reading assignments, online lectures, self-checks, quizzes, laboratory usage, post-testing, and assessment. Additionally a number of ancillary systems are slated for development. One major ancillary system is the scheduling system which allows students to schedule laboratory time by themselves using the user interface, thereby removing the scheduling load from laboratory staff. By far the most exciting ancillary system under development is the instructor interaction system. This system allows instructors to not only monitor live progress of a laboratory exercise but also to initiate problems or faults in the equipment such as having a circuit breaker fail, a transmission line trip out of service, or a generator governor fail. Through this system, dynamic situations requiring critical thinking can be developed for the PGL.

While the learning and ancillary systems were afterthoughts on the WebLab project, they should have been forethoughts. This is especially applicable to any new remote laboratory project. If remote laboratories are to take hold as a viable learning platform then time and resources need to be dedicated to developing all the learning and support materials needed to operate the laboratory. After all, would a classroom laboratory be bought or built without having learning materials and lesson plans to go along with it? Additionally, the data gathered by such systems will be invaluable in validating the effectiveness of remote laboratories in education.

The goal of integrating the learning and ancillary systems directly into the WebLab system is to create a one-stop-shop for any and all WebLab based laboratories. Once students enter the WebLab portal, all their needs are addressed. From one location students should be able to view online help, schedule laboratory time, view lectures, perform assignments, run laboratories, review their academic progress, and contact faculty and staff. This is also needed for instructors, as not all online instructors are based on location. Instructors will also need a one-stop-shop to track, monitor, assist, and assess their students’ progress. This is the ultimate goal of the WebLab, not to be just a remote access platform, but to become a truly comprehensive remote laboratory system.
Some of the information in this document may not be new, but it was acquired the hard way: through empirical experimentation. One of the unique properties of this project is that it was driven from an Energy Industry instrumentation and control perspective. While originally focused on the Energy Industry, an unexpected benefit of this project was the acquisition of new knowledge in many areas including but not limited to, online teaching and learning, research and development, technology integration into academia, and human machine interface development. This is the main motivating factor in producing this document. If remote laboratories are to become a permanent and legitimate part of online education, we need to learn from each other’s work. It is our hope that the information found in this document assists others in avoiding some remote laboratory pitfalls, while steering them to success.

Appendix A: WebLab System Details

WebLab POC V.2 (Generation 1):
Laboratory Equipment:
  7.5 Watt 120V AC Light Bulb
  25RPM 120V AC Motor
Control Equipment:
  National Control Devices R210 Serial Relay Controller, 2 – 120VAC 10AMP Omron Relays
Control Logic:
  Microsoft Visual Basic 6.0 Executable Residing on Server
Client Access System:
  Microsoft Visual Basic 6.0 ActiveX DLL Residing on Server
Web & Database Server:
  Microsoft Windows NT Server 4.0 running IIS and SQL Server
User Interface:
  Macromedia Flash 6.0 (current version is 8.0, Macromedia recently acquired by Adobe)
Video:
  ChillCam Video for Windows to JPG Encoder
  Logitech QuickCam

WebLab POC V.1

WebLab Version 2 (Generation 2):
Laboratory Equipment:
1 – Hampden Engineering Series 100 3 Phase AC Generator
1 – Hampden Engineering Series 100 DC Motor
1 – 3 Phase Circuit Breakers (Omron MK3P5SDC12 Relays)
1 – 240V DC Light Bulb Assembly

External Power Requirements:
120V AC @ 10A, 12V DC @ 2A

Control Equipment:
1 – IOtech DAQBoard/2001 – PCI DAQ Solution
1 – IOtech DBK2 – 4 Channel Analog Voltage Output Card
1 – IOtech DBK80 – 16 Channel Differential Analog Voltage Input Card

Transducers & Controllers:
1 – Camille Bauer Sineax U539 0-150V AC Voltage Transducer
1 – Camille Bauer Sineax M563 Multifunction Transducer (WATT, VAR, HZ)
1 – US Digital E6S Series Shaft Mounted 1024 CPR Optical Encoder
1 – US Digital ETACH Encoder to Analog RPM Converter (discontinued replaced with ETACH2)
1 – Minarik PCMXP05-115 PWM Controller (Generator Prime Mover)
1 – Minarik PCMXP02-115 PWM Controller (Generator Excitation)
3 – 0-5A AC Current Transducers
1 – Custom Built Darlington Transistor based Relay Controller
1 – Solid State Relay

Control Logic:
Microsoft Visual Basic .NET 2003 Service Residing on Server using IOtech DaqCOM API

Client Access System:
Microsoft Visual Basic 6.0 ActiveX DLL Residing on Server

Web & Database Server:
Microsoft Windows 2003 Server running with IIS and SQL Server 2000

User Interface:
Macromedia Flash 7.0 (current version is 8.0, Macromedia recently acquired by Adobe)

Video:
Axis 2100 Network Camera with Embedded Web Server (discontinued, replaced by AXIS 210)

WebLab Version 2

WebLab Version 3 (Generation 3):
Laboratory Equipment:
2 – Hampden Engineering Series 100 3 Phase AC Generator
2 – Hampden Engineering Series 100 DC Motor
1 – 3 Phase 120 to 240V AC Transformer Bank
2 – 3 Phase 240 to 120V AC Transformer Banks
1 – 3 Phase 240 to 24V AC Transformer Bank
2 – 3 Phase Resistive Loads (Wye Connected Light Bulbs)
1 – 3 Phase Capacitive Load (Wye Connected Capacitors)
1 – 3 Phase Inductive Load (Wye Connected Induction Motors)
20 – 3 Phase Circuit Breakers (Omron MK3P5SDC12 Relays)
14 – 3 Phase Disconnects (Omron MK3P5SDC12 Relays)
1 – 12VDC Variable Load (Model Train Set)

External Power Requirements:
120V AC @ 20A, 12V DC @ 5A

Control Equipment:
1 – Opto22 SNAP-UP1-M64 – 64 Channel PLC
1 – Opto22 SNAP-ENET-S64 – 64 Channel PLC
10 – Opto22 SNAP-ODC5SRC – 4 Channel Output Card (5-60VDC)
8 – Opto22 SNAP-AIV-4 – 4 Channel Analog Input Card (+-10VDC)
3 – Opto22 SNAP-AOV-25 – 2 Channel Analog Output Card (0-10VDC)

Transducers & Controllers:
5 – Camille Bauer Sineax U539 0-150V AC Voltage Transducer
5 – Camille Bauer Sineax U539 0-300V AC Voltage Transducer
2 – Camille Bauer Sineax M563 Multifunction Transducer (WATT, VAR, HZ)
1 – Camille Bauer Sineax G537 Phase Angle Transducer
2 – US Digital E6S Series Shaft Mounted 1024 CPR Optical Encoder
2 – US Digital ETACH Encoder to Analog RPM Converter (discontinued, replaced with ETACH2)
2 – Minarik PCMXP05-115 PWM Controller (Generator Prime Mover)
2 – Minarik PCMXP02-115 PWM Controller (Generator Excitation)
1 – Minarik LV02-24DC PWM Controller (Variable Load)
10 – 0-5A AC Current Transducers
6 – Solid State Relays

Control Logic:
Opto22 ioProject Based Programming Run from PLC Firmware

Database Interface:
Opto22 OptoProxy - PLC to Database Proxy Server

Client Access System:
Microsoft Visual Basic 6.0 ActiveX DLL Residing on Server

Web & Database Server:
Microsoft Windows 2003 Server running with IIS and SQL Server 2000

User Interface:
Macromedia Flash Professional 8.0 (Macromedia recently acquired by Adobe)

Video:
3 - Axis 2100 Network Camera with Embedded Web Server (discontinued, replaced by AXIS 210)
Acknowledgements

The author would like to acknowledge the following groups and people for their support and input on the WebLab Power Grid Laboratory project.

Bismarck State College (BSC):
- Administration: for their continued support of the WebLab concept.
- Faculty & Staff: for their assistance developing and building the WebLab PGL.

National Science Foundation:
- Course Curriculum and Laboratory Improvement: for the funding that made the WebLab PGL possible.
  - Award #DUE-0340927: May 1, 2004 – April 30, 2006: Award Amount $70,487.
- Russell Pimmel: for his continued support and advice throughout the WebLab project.

Western Area Power Administration’s Electric Power Training Center:
- Dennis Schurman: for his support of BSC and the WebLab project.
- Stan Adcock, Charlie Gray, Gary Zevenbergen, and Brad Nickell: for their support and technical expertise in the areas of electrical engineering, control systems, and power systems operation used in designing the WebLab PGL.

Western Area Power Administration Upper Great Plains Region:
- Robert Markel: for his assistance in power systems operation used in designing the WebLab PGL.

Western Area Power Administration Rocky Mountain Region:
- Brian Dunsmore: for his assistance in power systems operation used in designing the WebLab PGL.

Electrical Transmission Systems Technology (ETST) Advisory Board:
- Advisory Board Members: for their Electrical Industry input used in designing and building the WebLab PGL.

The author would also like to thank those too numerous to mention for their input and positive impact throughout the WebLab PGL project.
References

5. CHILLCAM Software, “ChillCam” [online product information]; available from http://www.chillcam.com/; INTERNET.
