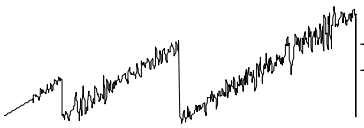
CU-NEES-08-08



NEES at CU Boulder

01000110 01001000 01010100

The George E Brown, Jr. Network for Earthquake Engineering Simulation

LTOS

LIGHTWEIGHT TELE-OPERATION SYSTEM MANUAL

Version 1.0

By

Kent Polkinghorne, I/T Manager
Robb Wallen, Instrumentation Engineer

September
2008

Center for Fast Hybrid Testing
Department of Civil Environmental and Architectural Engineering
University of Colorado
UCB 428
Boulder, Colorado 80309-0428

Table of Contents

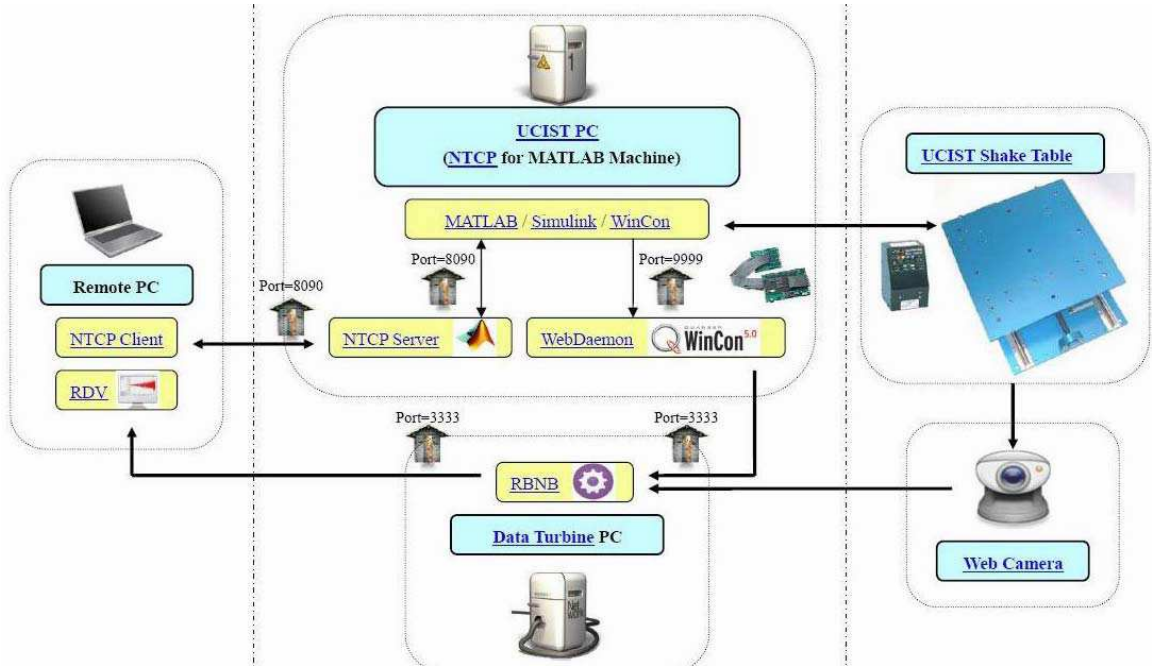
1 Overview

Many programs for remote operation and/or sensing of an experiment are available today. Some examples are LabVIEW Remote Panel (http://sine.ni.com/nips/cds/view/p/lang/en/nid/11017), Remote Desktop (http://www.microsoft.com/windowsxp/using/mobility/default.mspx), and RDV (http://it.nees.org/software/rdv/index.php). We note some drawbacks of these programs such as a large download size (about 100MB in the case of LabVIEW Remote Panel), non-intuitive interface, a complex information transfer protocol, and loss of control of the host. The Lightweight TeleOperation System (LTOS) is intended to overcome the drawbacks mentioned. The client is written in Adobe® Flash® which is already installed on millions of computers, the size of the download is very compact, the protocol is simple, and the host system can control the inputs and usage. Use of LTOS could include many Education and Outreach situations where individual users or schools would want to connect to an experiment hosted by a NEES site without needing extensive preparation on either end. Our version of LTOS is implemented with a LabVIEW server as the web host and a virtual instrument (VI) as the experiment's data host and data acquisition.

2 Comparing LTOS and RDV

We note some efforts that have already been undertaken for remote operation using RDV. A similar product to LTOS is The University Consortium of Instructional Shake Tables (UCIST) which was developed by Prof. Shirley Dyke to enhance university education in

earthquake engineering (S.J. Dyke, Z. Jiang, R. Christenson, X. Gao, and S. Courter, "Teleoperation and Teleparticipation of Instructional Shake Tables Using the NEES Cyberinfrastructure" Proceedings of the World Forum on Smart Materials and Smart Structures Technology, Chongqing and Nanjing, May 22-27, 2007). Their work led to a method for remotely viewing and operating a shake table using some tools from the NEES cyber infrastructure. For client viewing it relies on RDV with a custom display within RDV for experiment control. Below is a schematic of their implementation at the University of Connecticut:



(From "Teleoperation and Teleparticipation of Instructional Shake Tables Using the NEES Cyberinfrastructure")

Let us compare some pros and cons between RDV and LTOS:

RDV	LTOS
Contains a technical interface meant for engineers with many options and buttons	Interface only has what is needed, simple for non-engineers to operate
Not always in real-time mode, difficult for novice users to tell if they are looking at the past or the now	Always real-time display
Layout is limited to grid, not remotely changeable during session	Layout more precisely controlled by server, components can be created and deleted during the session by the server
Real-time graph data	Quick real-time display, smooth-scrolling graphs
Requires Ring Buffer Network Bus for communication from experiment to client	Simple network protocol, easy to implement into existing control software, fewer layers mean quicker feedback

No ability to control the experiment without additional software	Built-in controls
Already exists, mature product	Just starting out, not in use by as many users
More scientific (data pause and replay, user can configure channels monitored)	Less scientific but can still save data results to user's computer for analysis

In general the focus for LTOS is the non-engineering community but is not limited to it. Some examples would include grade school teachers who want to do a live demonstration in their class, university students taking introductory classes needing to run a small centrifuge, high school honors students remotely operating a fan to see the effects on a miniature power-generating windmill.

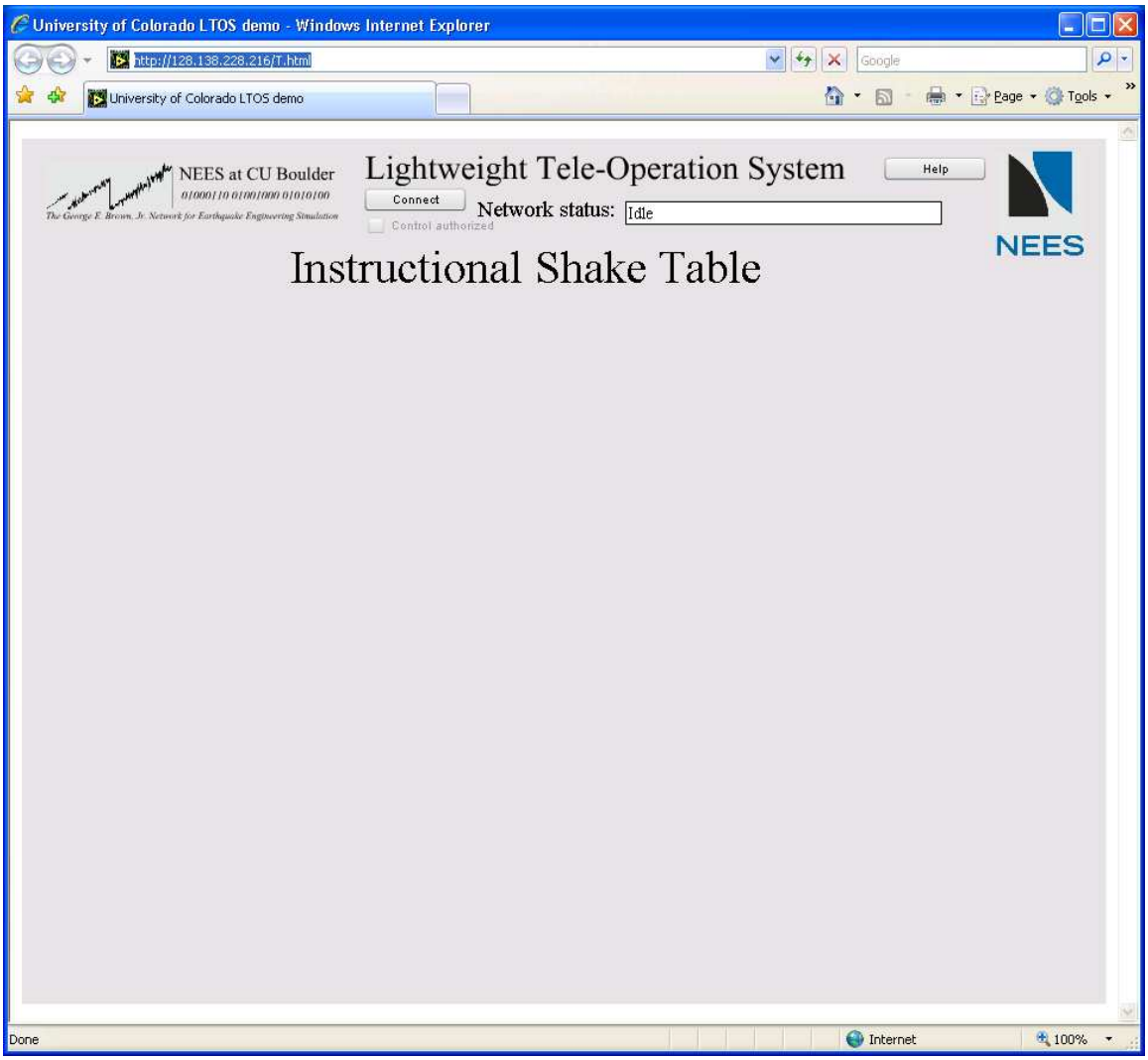
3 Quick set-up with demonstration files

A simple demonstration can be done if you've downloaded these sample files:

Teleoperation client.swf	The Flash client itself.
Hosting_page.html	A sample HTML source that hosts the Flash Client.
AC_RunActiveContent.js	An Adobe file used by *Hosting_Page.html* to facilitate the opening of the Flash client.
LTOS\Shaker.vi	The main VI. It communicates with *Server.vi* and operates the shaking table.
LTOS\Server.vi	Handles connections and the data transfer to/from the client.
LTOS\EOT custom.txt	A customized ground motion that impresses young viewers of our shaking table.
LTOS\buildLayoutCommand.vi	In the future this will be called by *Shaker.vi* to construct the actual layout messages sent to the Flash client.
LTOS\buildUpdateCommand.vi	Is called by *Shaker.vi* to construct the actual update messages sent to the Flash client.
LTOS\processRequest.vi	Is called by *Shaker.vi* to interpret the update messages received from the Flash client.
LTOS\TCPConnection.vi	Is called by *Shaker.vi* to handle the details of TCP connections.

1. Edit *Hosting_page.html* and replace the text "neesLTOS.colorado.edu" with the IP address of the computer running the server VIs.
2. Open the files *LTOS demo.vi* and *Server.vi* in LabVIEW.
3. Enable the LabVIEW web server if it isn't already. Do so by going to Tools / Options / Web Server: Configuration / Enable Web Server. The web server root needs to be *C:\Program Files\National Instruments\LabVIEW 7.1\www.*
4. Start both VIs (order doesn't matter).

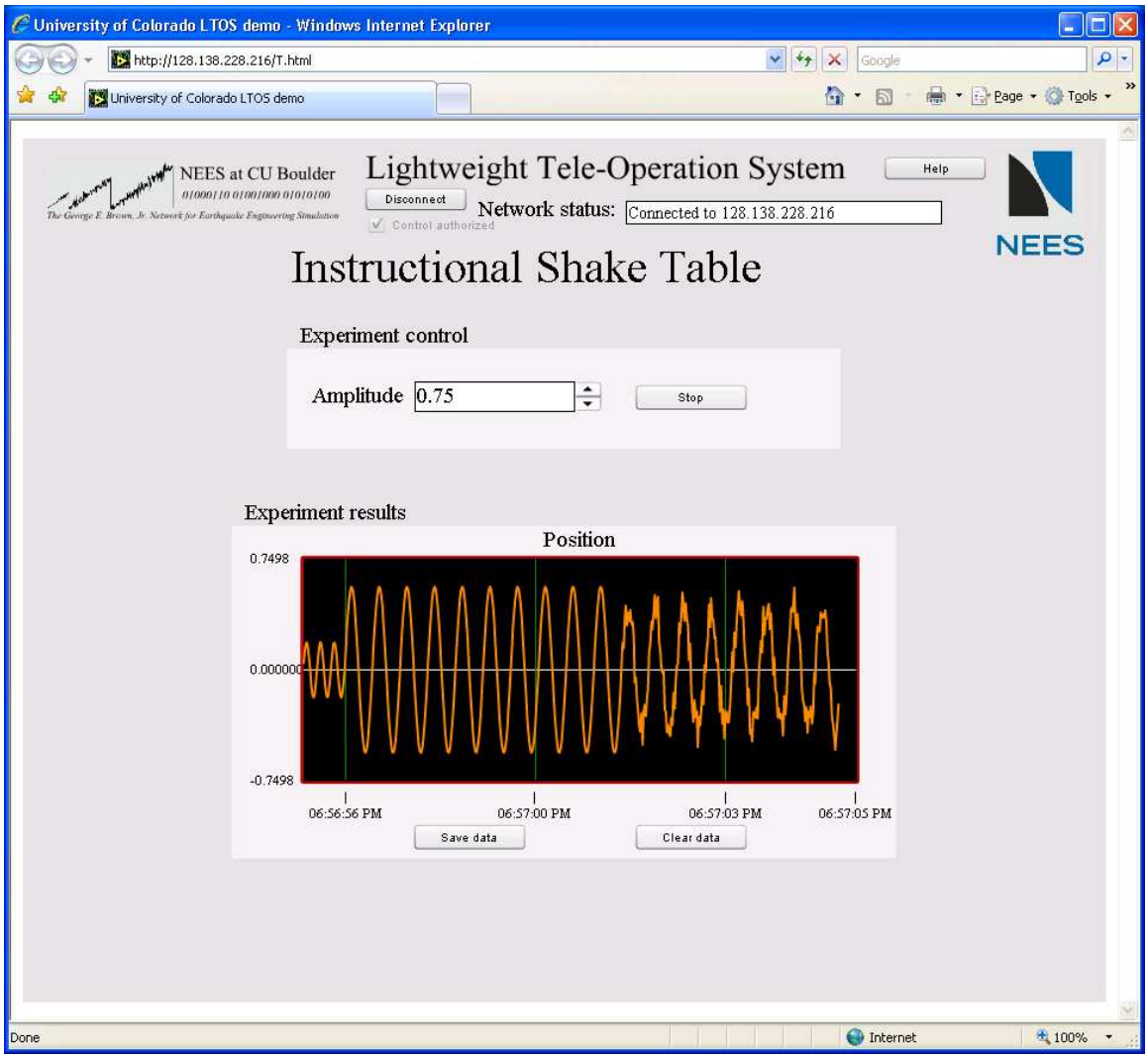
5. Browse to `http://<LabVIEW's IP address here>/Hosting_page.html` and you will see the LTOS client.



The Flash client loaded but not connected to the LabVIEW server.

6. Click on "Connect" to have the client connect to LabVIEW.
7. You can now play with the controls on the client.

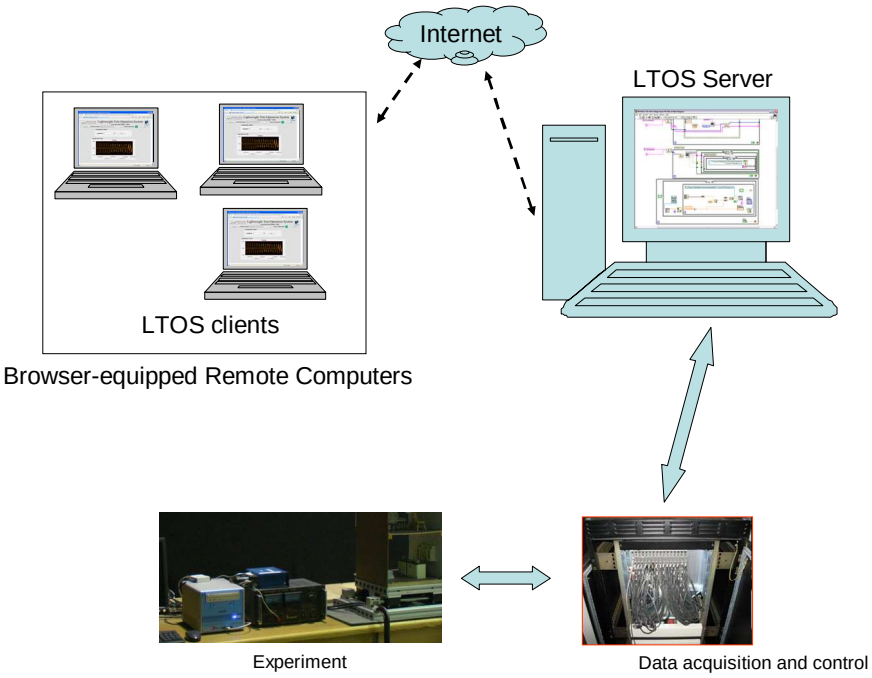
This LabVIEW server is similar to what we use on our Education and Outreach shaking table that students can watch. The Flash client when used with the demonstration server should look like this:



The Flash client when connected to the demonstration server and with the experiment running.

4 Communication path

LTOS uses comparatively few levels of communication. An example path of communication is shown below. LTOS requires no middleware between the software controlling the experiment and the client. Our demonstration LabVIEW program does the data acquisition and the LTOS communication together.



A schematic of the data flow in a typical LTOS configuration

5 Client configuration

5.1 How to host

Hosting the client itself is very easy. The files *AC_RunActiveContent.js* and *Teleoperation client.swf* are placed into a directory accessible by a web server. A HTML source file also needs to be present that would be written and customized by the individual site. Some HTML code to magically make the Flash client appear needs to be on that page (the code examples are listed in the next section). That's it!

It is possible to host the Flash client on a server different than the server hosting the experiment data.

5.2 HTML tags

Because of the variety of browsers, Adobe recommends quite a few HTML tags in the page hosting the Flash client to achieve maximum usability. This includes tags for a JavaScript launcher, for ActiveX, and for the Netscape/Mozilla-style <embed> method. The Adobe default is to use all of the three methods in the same hosting page, see the example hosting page. A specification of HTML can be found at http://www.w3.org/TR/html401/.

Some individual variables that can be set via the tags and their function are:

1. `FlashVars`: this is a general Flash variable and we use it to pass in the name and port of the data server, the URL for the help button, and the title to be displayed in the client. It takes the format

"Server=neesLTOS.colorado.edu&Port=3688&HelpURL=http://nees.colorado.edu/&ClientTitle=Instructional%20Shake%20Table" where neesLTOS.colorado.edu is the name of our demonstration server, 3688 is the TCP port, the help button navigates to http://nees.colorado.edu/, and the title displayed in the client is "Instructional Shake Table".

2. `width, height`: usually both are "100%" but can be changed to any percentage to cause the Flash client to consume more or less of the visible window.
3. `align`: usually "middle", determines how the client is located within the browser window. Can also be "L", "R", "T", or "B" to align along the left, right, top, or bottom edge.
4. `scale`: usually "showall", determines if the client should scale down if the visible window is made smaller. Can also be "noborder" which might crop an edge or "exactfit" which will change the original aspect ratio.
5. `bgcolor`: usually "#ffffff", and is the HTML color code for the background of the client. Use to make the client's background match that of the hosting page.

Note that these variables are all listed in each of the three tag methods so if you change one of them you need to change it in all three sets of tags. The sets are listed in the next section.

5.2.1 JavaScript tag example

```
<head>
<script language="javascript">AC_FL_RunContent = 0;</script>
<script src="AC_RunActiveContent.js" language="javascript"></script>
</head>
<body>
<script language="javascript">
  if (AC_FL_RunContent == 0) {
    alert("This page requires AC_RunActiveContent.js.");
  } else {
    AC_FL_RunContent(
      'codebase',
'http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#version=9,0,0,0',
      'width', '100%',
      'height', '100%',
      'src', 'Teleoperation client',
      'quality', 'high',
      'pluginspage', 'http://www.macromedia.com/go/getflashplayer',
      'align', 'middle',
      'play', 'true',
      'loop', 'true',
      'scale', 'showall',
      'FlashVars',
'Server=neesLTOS.colorado.edu&Port=3688&HelpURL=http://nees.colorado.edu/&ClientTitle=Ins
tructional%20Shake%20Table',
      'wmode', 'window',
      'devicefont', 'false',
      'id', 'Teleoperation client',
      'bgcolor', '#ffffff',
      'name', 'Teleoperation client',
      'menu', 'false',
      'allowFullScreen', 'false',
      'allowScriptAccess','sameDomain',
      'movie', 'Teleoperation client',
      'salign', ''
      ); //end AC code
  }
</script>
```

5.2.2 ActiveX tag example

```
<object classid="clsid:d27cdb6e-ae6d-11cf-96b8-444553540000"
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#version=9,0
,0,0" width="100%" height="100%" id="Teleoperation client" align="middle">
```

```
<param name="FlashVARS" value="
Server=neesLTOS.colorado.edu&Port=3688&HelpURL=http://nees.colorado.edu/&ClientTitle=Inst
ructional%20Shake%20Table " />
<param name="allowScriptAccess" value="sameDomain" />
<param name="allowFullScreen" value="false" />
<param name="movie" value="Teleoperation client.swf" />
<param name="menu" value="false" />
<param name="quality" value="high" />
<param name="bgcolor" value="#ffffff" />
```

5.2.3 Netscape/Mozilla tag example

```
<embed FlashVARS="
Server=neesLTOS.colorado.edu&Port=3688&HelpURL=http://nees.colorado.edu/&ClientTitle=Inst
ructional%20Shake%20Table " src="Teleoperation client.swf" menu="false" quality="high"
bgcolor="#ffffff" width="100%" height="100%" name="Teleoperation client" align="middle"
allowScriptAccess="sameDomain" allowFullScreen="false" type="application/x-shockwave-
flash" pluginspage="http://www.macromedia.com/go/getflashplayer" />
```

5.3 Flash version

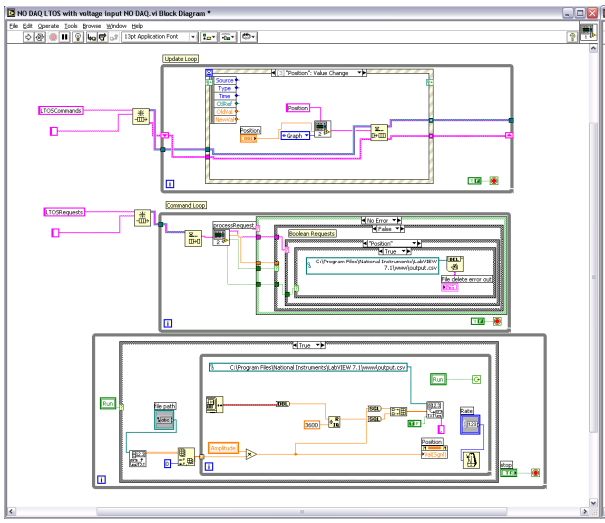
The LTOS Flash client was developed using Adobe Flash CS3 Professional version 9.0. The client contains ActionScript 3.0 and needs to have Flash version 9.0 or better. The current version is 9.0.124.0 and is already installed on many computers at this time.

6 Demonstration server configuration

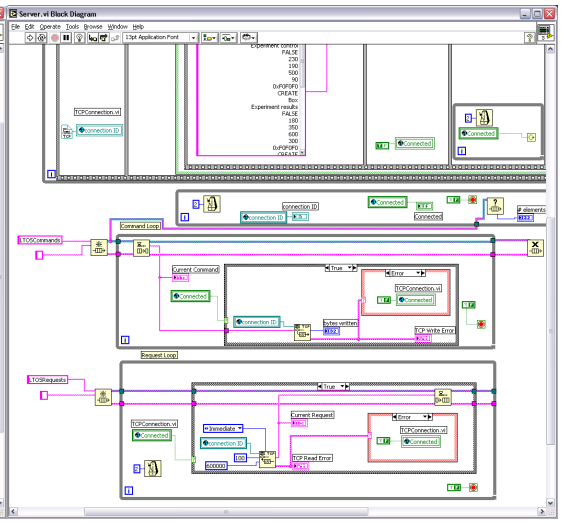
The included demonstration LabVIEW server is an adaptation from our Education and Outreach shaking table to show how one could use LabVIEW as a server to the LTOS client. The server could be implemented in a variety of programming languages and operating systems as long as it can receive incoming TCP connections and send and receive text packets. The protocol was kept simple to allow for the easy addition to existing control programs of the LTOS server.

The demonstration server presents three controls to the client. The first is the run/stop button which starts and stops playback of the *EOT custom.txt* file. The second allows the user to change the amplitude of the playback. The third is the graph which displays the playback versus time. Normally one would see something such as a daqMX control in LabVIEW to control an actuator's movement from the playback for example but we've removed that for this demonstration.

After following the directions for doing a quick set up of the demonstration server and connecting with the client, one would do something like click the Run button. A position is displayed on the graph and the user can change the amplitude to see the results. If LabVIEW's web server is enabled and set to the default directory of *C:\Program Files\National Instruments\LabVIEW 7.1\www* then clicking on "Save data" in the client will download a file named "output.csv" which is being written by the server. The demonstration server won't operate if this directory doesn't exist. Some parts of the LabVIEW demonstration server are shown below.



The LabVIEW experiment control code.



The LabVIEW communication server code.

7 Protocol

7.1 General message format

The messages transferred back and forth from the Flash client to the data server are simple to implement and use. They are asynchronous in that no message depends on a response and no message causes a blocking action. Each message is composed of one or more lines of text with a newline (\n) at the end of each line and a NULL (\00) at the end of each message. Our example LabVIEW server assumes that each message will arrive as one TCP packet (without fragmentation or combining). Even though TCP doesn't guarantee this the LabVIEW server hasn't had problems with it. The Flash client can tolerate fragmented or combined TCP packets.

The example server listens on TCP port 3688 and the client is currently hardwired to connect to this port. After the server receives a connection it sends the commands to set up the layout of controls on the client. After the client receives a layout command it sends a message setting the "SETUP" variable to TRUE which is interpreted by the server to send the current values of all variables to the client. This is the mechanism by which the current values in the LabVIEW server are communicated to the client.

7.2 Layout messages

The messages in this group are sent from the server to the client to configure the position and type of controls. Controls can be created at any time and can also be destroyed at any time thereby allowing for different "screens" on the client during one session.

Attributes to consider for every control to be created are what type it will be, its name, its X and Y position, and if it is an input or output (or both). As far as the Flash client is concerned, every control is both readable and writeable (an indicator and a control in LabVIEW terms). However a flag in the layout sequence determines whether the Flash client will allow updating or not of a particular control.

Most layout commands follow the same order and look like this (to create a numeric input named "Acceleration" that is not changeable by the user, positioned at (100, 350), and has lower and upper limits of [0,2]):

```
CREATE
Numeric
Acceleration
FALSE
100
350
0
2
```

The mandatory parameters are the control's name, whether it is changeable by the user or not, it's X position, and its Y position. Type-dependant parameters may follow and are listed below. Remember that the message above would be terminated by a NULL after the last carriage return (and must not have another carriage return following the NULL). The available control types are:

- `ToggleLight`: similar to the Boolean indicator in LabVIEW, it looks like a light that is either dark for FALSE or green for TRUE.
- `ToggleSwitch`: similar to the Boolean control in LabVIEW, it is a toggling two-button switch. It requires one more parameter which is the title of the off position's button. The top position sends TRUE and the bottom sends FALSE.
- `ToggleButton`: similar to `ToggleSwitch`, but it a single button with a label that toggles between two values. It requires one more parameter which is the title of the button to activate the FALSE position.
- `Numeric`: similar to a numerical indicator/control in LabVIEW, it is a text box where the user can type a number or click up/down buttons (or use the mouse wheel) to change the value. It requires two more parameters which are the minimum and maximum values that the user can enter.
- `Textual`: a standard text box where the user can enter text or messages can be shown.
- `Box`: a box to draw on the screen. It requires two more parameters which are the width and height.
- `Graph`: a type of plot where each horizontal pixel represents one data point. The plot scrolls when it is filled and has automatic ranges for the Y axis and no labels on the X axis. The user can click on the graph to clear it which can send a message to the server.
- `GraphTimed`: similar to `Graph` but has labels on the X axis as well showing the time at various points.
- `XYseries`: similar to `GraphTimed` but plots an X versus Y line for the last 500 data points. It has labels on both the X and Y axis and automatically scales to the largest of all data received since resetting the plot.

7.3 Information messages

Information messages are what convey data changes, either from the experiment or from the user, between the server and client. The message format is the same whether the server or the client sends it and is similar in design to the layout messages.

A sample information message looks like this (if the server were to change the numeric value of Acceleration to 1.22 and deactivate the light named Shaking):

```
UPDATE
Acceleration
1.22
Shaking
FALSE
```

Any number of control updates can be combined into one message even though our LabVIEW demonstration server doesn't do that. The XYseries and GraphTimed controls require two parameters separated by a semi-colon in their update, such as in this example (to update the Temperature graph with a Y value of 27.4 and an X value of 578):

```
UPDATE
Temperature
27.4;578
```

Remember that each of the above messages is terminated with a NULL.

The Boolean types use the strings "TRUE" and "FALSE". TRUE is associated with an on or activated state, and FALSE with an off or deactivated state. For the graphing types when the user clicks the "Clear data" button (or clicks on the graph background) then an update is sent to the server with the TRUE value. In our demonstration server this action deletes the data file. When a user clicks on "Save data" a HTTP download is requested from the server to download the file "/LTOS/output.csv".

7.4 Flash network security policy file

Flash was designed to be very secure and impossible to use for writing a virus or other malware. To that end, Flash attempts to connect to the server hosting the experiment data to download a security policy file which Flash uses to know if it should allow further connections. Flash attempts to connect on TCP port 843 and sends the string "<policy-file-request/>" terminated by a NULL. The server should answer this request with a policy file similar to this one:

```
<?xml version="1.0"?>
<!DOCTYPE cross-domain-policy SYSTEM "http://www.adobe.com/xml/dtds/cross-domain-
policy.dtd">

<cross-domain-policy>
        <site-control permitted-cross-domain-policies="master-only"/>
        <allow-access-from domain="*" to-ports="3688"/>
</cross-domain-policy>
```

The "to-ports" section would need to be modified to include the port you are using to host the experiment data.

7.5 A note about TCP/IP communication and firewalls

The LTOS client for our demonstration uses TCP ports 80, 843, and 3688. Ports 80 and 3688 could be changed to anything the site wishes to use but we've had limited success in using a different port besides 843 for the Flash security policy file transfer.

Most organizations today employ a firewall to prevent outsider access to individual computers on the network. If a site is intending to allow access from the Internet via the LTOS client then they need to address opening these three ports in the firewall which is often made in a request to the IT department serving the organization.

8 Future work

During the development of LTOS version 1.0 several ideas came about for future work. Some of these ideas are to improve deficiencies in the current version and others are ideas of things that would be beneficial to have some day.

1. Add parameters to the graph controls to change the filename used for saving, the history length, graph size, and to control fixed scaling.
2. Add an authentication scheme between the Flash client and server (such as requiring a password before allowing control).
3. Overcome local web browser caching of the graph's saved file (currently the web browser will sometimes ignore new data).
4. Modify the LabVIEW demonstration server so that the layout commands are send from the shaker VI (using the buildLayoutCommand VI) instead of as a fixed string from the server.
5. Create the ability to allow multiple clients to connect for viewing and for one client at a time to be authorized to control the experiment.
6. Embed a video feed into the Flash client (possibly using *ffmpeg* and the Axis video servers).
7. Develop a MATLAB/Simulink experiment server that communicates with LTOS.

9 Acknowledgements

This work was made possible by funding support from the National Science Foundation through a sub-award to the Network for Earthquake Engineering Simulation (NEESinc).