A Web-based Interface
For Model Railroading

Kevin Sendra
December 12, 2002
**Project Summary**

The Engineering Department at Bradley University requires all undergraduate electrical engineering majors to complete a year long project. This report is about a project to design a web-based interface for a model train controller. The interface will consist of a client and a server, both written in Java. The user of the client will be able to control train speed and direction, switch positions, and any lighting or other features incorporated into the layout. The client and server will be discussed in more detail below.

**Detailed Project Description**

The project consists of three main parts. These parts are the client, the server, and the webcam. Each of these will be looked at separately in the following paragraphs. A block diagram of the system is shown below.

![General Block Diagram](image)

Figure 1: General Block Diagram

The client will interface with the user. This will be a graphical user interface (GUI) written in Java and included in an applet that will be accessible though a web page. The interface will include: a webcam image, a slider to control the train speed and direction, a picture of the layout for the user to click to change switch positions, a text area to show layout status information and any error messages, and an emergency stop button to stop the train from moving. The client will send the commands received from the user to the server over a TCP/IP connection. The inputs and outputs are shown in figure 2.
The server will be located on the personal computer near the layout. This computer will include the controller application of the project this interface is being written for. The server will communicate between this controller application and any clients connected. The server will be set up to handle multiple client connections. One connection will be able to control the train, any other connections can wait to control the train or view the webcam of the layout. The server will also be written in Java, but this will only run as an application, not as an applet. The server doesn’t have a real user interface; its purpose is just to connect the client and the controller application. Since the controller application will not be able to connect directly to this, the server will write to a file that the controller application can read from. The inputs and outputs can be seen in Figure 3.

There last portion of this project is a commercial webcam. The webcam will provide pictures of the layout to the local computer. From there, a program will send these images to users connected to the server. Video streaming will be the preferred method of transfer for the images. More research will be done to determine the best and most cost efficient way to send these images.

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Commands ‡</td>
<td>‡ Web Cam Image</td>
</tr>
<tr>
<td>Communication from Server ‡</td>
<td>‡ Feedback to user</td>
</tr>
<tr>
<td>Video Feed ‡</td>
<td>‡ Communication to Server</td>
</tr>
</tbody>
</table>

Figure 2 Inputs and Outputs from the Client
<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feedback from Controller Application</td>
<td>Commands to Controller Application</td>
</tr>
<tr>
<td>Web Cam Image</td>
<td>Communication to Client</td>
</tr>
<tr>
<td>Communication from Client</td>
<td></td>
</tr>
</tbody>
</table>

Server

Figure 3: Inputs and outputs relating to the server

- **Server Ready/Not ready for connection**
  - Try to establish read and write connections with server
    - Read Data about layout from server
      - Display error
    - Return in use or error messages
  - Wait for command
    - Time out, close connection
      - Server open for new connections

- **If Error Occurs**
  - If Catastrophic Error
  - If Minor Error

- **Client sends a command that server receives**
  - Server writes command to file for Controller Application
    - Client requests new image
      - Server sends image to client
        - Client reads error and displays to user
      - Server sends error information
        - Client disconnects

Figure 4: Flowchart for the client and server (subject to change)
Research and Work Completed

The cost of the project is pretty minimal. The PC to run the interface on will not add cost as it is necessary for the local control. The only cost is from the webcam. Webcams usually cost about 20-30 dollars which is inexpensive.

The software will be written using Java. Java is a free programming language and is very common on the internet. I’ve had some experience using Java in a web programming course I took prior to this project.

I have already written a simple server and client to familiarize myself with Java again. This server can send information to the client. There isn’t a real graphical user interface; it’s just a box you can type in and then click a button to send to the server. I will build around this code to implement the rest of the features mentioned previously. The code I have written can be seen in appendix B.

Schedule of Work to be Completed

December and January – Continue work on the server and client in Java. Begin work on the graphical user interface.

February – Start working on the webcam and webcam interface.

March – Finish work on remaining features of the web interface.

April – Help the other project if necessary or begin testing the interface with the layout (hopefully testing will take place before this as features are added).
Standards Used

The only two standards used are the Java programming language and its TCP/IP connection sockets.

Equipment List

- Personal Computer to run server on (same as controller application uses)
- Webcam at an estimated cost of about $30.
- A java compiler from java.sun.com for free.
- Computer and web browser for client end (required by user)

References

References used are:

- [http://cegt201.bradley.edu/~olekmali/courses/02f-ee409/examples/java/default.htm](http://cegt201.bradley.edu/~olekmali/courses/02f-ee409/examples/java/default.htm)
- “Core Web Programming” by Marty Hall