David M. Zeff (S.B. #63289)
Law Offices of David M. Zeff
1388 Sutter St., Suite 820
San Francisco, CA 94109
Telephone: (415) 923-1380
Facsimile: (415) 923-1382
ZeffLaw1@aol.com

Attorneys for Defendant
Kevin Russell

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

SAN FRANCISCO DIVISION

ROBERT JACOBSEN, Plaintiff, vs. MATTHEW KATZER, KAMIND ASSOCIATES, INC., and KEVIN RUSSELL, Defendants.	Case No. C 06 1905 JSW **DECLARATION OF MATTHEW KATZER IN SUPPORT OF SPECIAL MOTION BY DEENDANT KEVIN RUSSELL TO STRIKE COUNTS 5 AND 7 IN THE COMPLAINT [Cal. Civ. Proc. Code Sec. 425.16]** Date: August 4, 2006 Time: 9:00 a.m. Dept: Courtroom 2, 17th floor Hon. Jeffrey S. White

I, Matthew Katzer, declare:

1. I am Chief Executive Officer and Chairman of the Board of Directors of Kamind Associates, Inc. ("KAM"). I am a named defendant in this action. If called as a witness, I would and could testify to the following as a matter of personal knowledge.

2. I am authorized by KAM to make this declaration in support of the motion by Kevin Russell to dismiss for lack of personal jurisdiction and make this declaration in my capacity as KAM's Chief Executive Office and Chairman of the Board of Directors.

3. KAM authorized Kevin Russell, as KAM's attorney, to send a FOIA request to the United States Department of Energy ("DOE") on behalf of KAM with regard to the DOE's apparent sponsorship of the java model railroad interface ("JMRI") project, which distributes model railroad software that KAM believe infringes on patents owned by KAM. A copy of this request is attached as Exhibit 1.

4. My belief that the DOE sponsored the JMRI project was based on the following facts, among others:

a. I was aware of other instances in which governmental agencies sponsored types of open source software. For example, the United States Navy, through the Office of Naval Research and the Naval Surface Warfare Center, sponsored a conference and paper on real-time software controllers for digital model railroad systems in 1993. Additionally, the National Science Foundation funded research for a paper entitled "A Laboratory Platform to Control a Digital Model Railroad over the Web using Java" (undated). Copies of both papers are attached to this declaration as group Exhibit 2.

b. I believe that I saw notice of the formation of JMRI on DOE or other official letterhead on a JMRI users support website (http://groups.yahoo.com/group/JMRIusers) in about January or February of 2004.

c. I have found and downloaded not fewer than 2,320 documents promoting JMRI from an e-mail account at lbl.gov, including requests for funding. Copies of representative documents are attached as group Exhibit 3.

d. By performing a search on the internet, I determined that the lbl.gov email address was associated with the DOE.

5. The purpose of the FOIA request was to obtain any publically available information subject to disclosure under the FOIA about activities apparently sponsored by DOE that appeared hostile to KAM and its interests. It was also intended to caution DOE that KAM regarded some of the JMRI software as infringing KAM's patents.

6. It was not the purpose of the FOIA request to embarrass Jacobsen with his employer, as alleged in the complaint. At the time of sending the FOIA request, I had no knowledge of Jacobsen's employment status, either with the DOE or any other employer. At the time of sending the FOIA request, I assumed that Jacobsen worked for the University of California at Berkeley based on a representation Jacobsen made in an email posting to a model train internet newsgroup in 2001. A copy of this email is attached as Exhibit 4.

7. A purpose of the FOIA request was to gather information about the JMRI activity, which appeared to be conducted with the sponsorship of the Department of Energy, in the preparation of a lawsuit, contemplated in good faith, against JMRI participants for infringement of KAM's patents.

I declare under penalty of perjury under the laws of the State of California that the foregoing is true and correct.

Executed on May 9, 2006.

Matthew Katzer

Katzer Declaration: Exhibit 1

Katzer Declaration: Exhibit 1

CVMS

Law Offices
Chernoff, Vilhauer, McClung & Stenzel, LLP
Intellectual Property Law
Including Patent, Trademark, Copyright
and Unfair Competition Matters

1600 ODS Tower
601 S.W. Second Avenue
Portland, Oregon 97204-3157
Telephone: 503-227-5631
Fax: 503-228-4373

* Jacob E. Vilhauer, Jr.
* Dennis E. Stenzel
* Charles D. McClung
* Donald B. Haslett
* J. Peter Staples
* William O. Geny
* Nancy J. Moriarty
* Kevin L. Russell

Daniel P. Chernoff
(1935-1995)

* Tim A. Long
* Kurt Rohlfs
* Brenna K. Legaard
* Susan D. Pitchford
* J. Douglas Wells
Holly L. Bonar

* Registered Patent Attorney

David S. Fine
Senior Law Clerk

October 27, 2005

FOIA Officer
Office of Science
US. Department of Energy
1000 Independence Avenue, SW
Washington, DC 20585

Re: Freedom of Information Act Request for all Documents Related to Patent Infringement, Status, Funding, Distribution of Contributions and Management Practices Associated with the JMRI Project by Berkley Labs.
Our File No.: 7431.0081

Dear FOIA Officer:

This request for documents is made pursuant to the Freedom of Information Act, 5 U.S.C. §§ 552 *et seq*. This request is made on behalf of KAMIND Associates, Inc. ("KAM") and relates to information gathered by the Physics Division Berkley Livermore Labs (LAB) regarding their duties for JMRI project. KAMIND Associates, Inc. is a small software vendor that has patents being infringed by the JMRI project sponsored by the LAB.

Please consider documents to include all writings, memoranda, letters, notes, working papers, minutes of meetings, photocopies, data, graphs, charts, photographs, inspection reports, compliance reports, records, e-mails (sent, received or drafts), digitized voice communications and any other format of information regarding the JMRI program project (hereafter referred to as COMMUNCIATIONS).

equestr agreed to pay up to $5000.00 by voicemail on 11/8. Please notify if it will exceed.

F2005-00085

on the Yahoo Groups: JMRI USERS. [Exhibit 8, 9]
13 All COMMUNCIATIONS and logs from Skype ID: JacobsenRG related to JMRI

KATZER DECL. EXHIBIT 1

CVMS

FOIA Officer
Office of Science
U.S. Department of Energy
October 27, 2005
Page 2

Specifically, KAM requests:

1. Funding information from the Department of Energy for the JMRI program at the LAB.
2. All COMMUNCIATIONS about the JMRI program and proceedings from Robert Jacobsen (LBNL, 1 Cyclotron Rd, MS 50A2160, Berkley Ca, 94720 email address Bob_Jacocobsen@lbl.gov). **[Exhibit 1, 2, 3]**
3. Complete financial records and all COMMUNICATIONS from contributions (PAYPAL) to support the JMRI program. **[Exhibit 4, 5]**
4. Complete financial records verifying that the funds received for Government project JMRI were deposited in US Treasury.
5. Transcripts of communications (COMMUNCIATIONS) to any JMRI team members regarding KAM. **[Exhibit 6]**
6. All COMMUNCIATIONS regarding patent investigation of KAM. **[Exhibit 6]**
7. All draft COMMUNCIATIONS to any member of the JMRI development community. There are 18 members as of 10/24,2005. **[Exhibit 2]**
8. All COMMUNCIATIONS to any member of the JMRI_STRATEGY group. **[Exhibit 7]**
9. All COMMUNCIATIONS regarding legal opinion on Department of Energy personal at any locations regarding the JMRI activities.
10. All COMMUNCIATIONS from email archives (2000 to present) to any member on the Yahoo Groups: JMRI DEVELOPERS. **[Exhibit 2]**
11. All COMMUNCIATIONS from email archives (2000 to present) to any member on the Yahoo Groups: JMRI STRATEGY. **[Exhibit 7]**
12. All COMMUNCIATIONS from email archives (2000 to present) to any member on the Yahoo Groups: JMRI USERS. **[Exhibit 8, 9]**
13. All COMMUNCIATIONS and logs from Skype ID: JacobsenRG related to JMRI activities.
14. Copies (and all drafts) of the welcome letter from the LAB by Bob Jacobson to JMRI members posted on Yahoo groups and or source forge.
15. All information related to the KAM legal action in federal court.
16. All travel activities that relate to the JMRI activities during July 2001, 2002, 2003 and 2004 period.

KATZER DECL. EXHIBIT 1

17. All COMMUNICATIONS between the following Allen Bryne, Ralph Kimball, Michael Woodman, Graham Plowman, and Jerry Britton. **[Exhibit 10, 11, 12]**
18. All COMMUNICATIONS about JMRI Royalty payment of $203,000 for KAM Train Server licenses. **[Exhibit 13]**

Sincerely,

Kevin L. Russell

KLR:kk
Enclosures

cc: KAMIND Associates, Inc.
(w/enclosures)

Katzer Declaration: Exhibit 2

Katzer Declaration: Exhibit 2

IML lab Real-Time Digital Model Railroad Project

Proceedings of the IEEE Conference on Real-Time Applications
May 13-14, 1993
New York, N.Y.
Sponsored by: IEEE Computer Society, Office of Naval Research, Honeywell Incorporated and Naval Surface Warfare Center.

A Real-Time Software Controller for a Digital Model Railroad System*

by
Roger W. Webster, Ph.D.
David Hess
Intelligent Machines Laboratory
Department of Computer Science
Millersville University P
Millersville, PA USA 17551
(717) 872-3539
email: webster@cs.millersv.edu

Abstract

This paper describes a real-time software controller for a digital model railroad.
primitives of fork, pipe, and signal are used to perform interprocess communicatic
executing tasks, (1) a Scanning Task, (2) a Scheduler and Collision Avoidance task,
Interface (GUI) task. The software engineering objective of this real-time system i
digital locomotives each running on the same track layout while at the same time al
scheduling system to "run" the trains. The control software continuously monitors t
of each train's location and direction, and is constantly performing collision avoi
digitally encoded with a chipset that is addressable, therefore messy block wiring
unnecessary. Each digital locomotive and digital turnout switch responds to compute
address.

I. Introduction.

In this railroad layout there are six digital turnout switches, two digital locomot
to manage and control (Figure 1). The objective is to move the trains around the tr
to the scheduling algorithm without collision. The fifteen reed contact sensors are
around the track (Figure 2). Magnets are attached to each locomotive which trip ree
implanted in the track. This configuration provides an interesting, experimental pl

real-time systems for undergraduates in Computer Science and Computer Engineering. While many undergraduate courses in Real-Time Systems acquaint students with the fu

* This work partially funded by the National Science Foundation (NSF) under Grant N Millersville University and by the Faculty Grants Committee of Millersville Univers

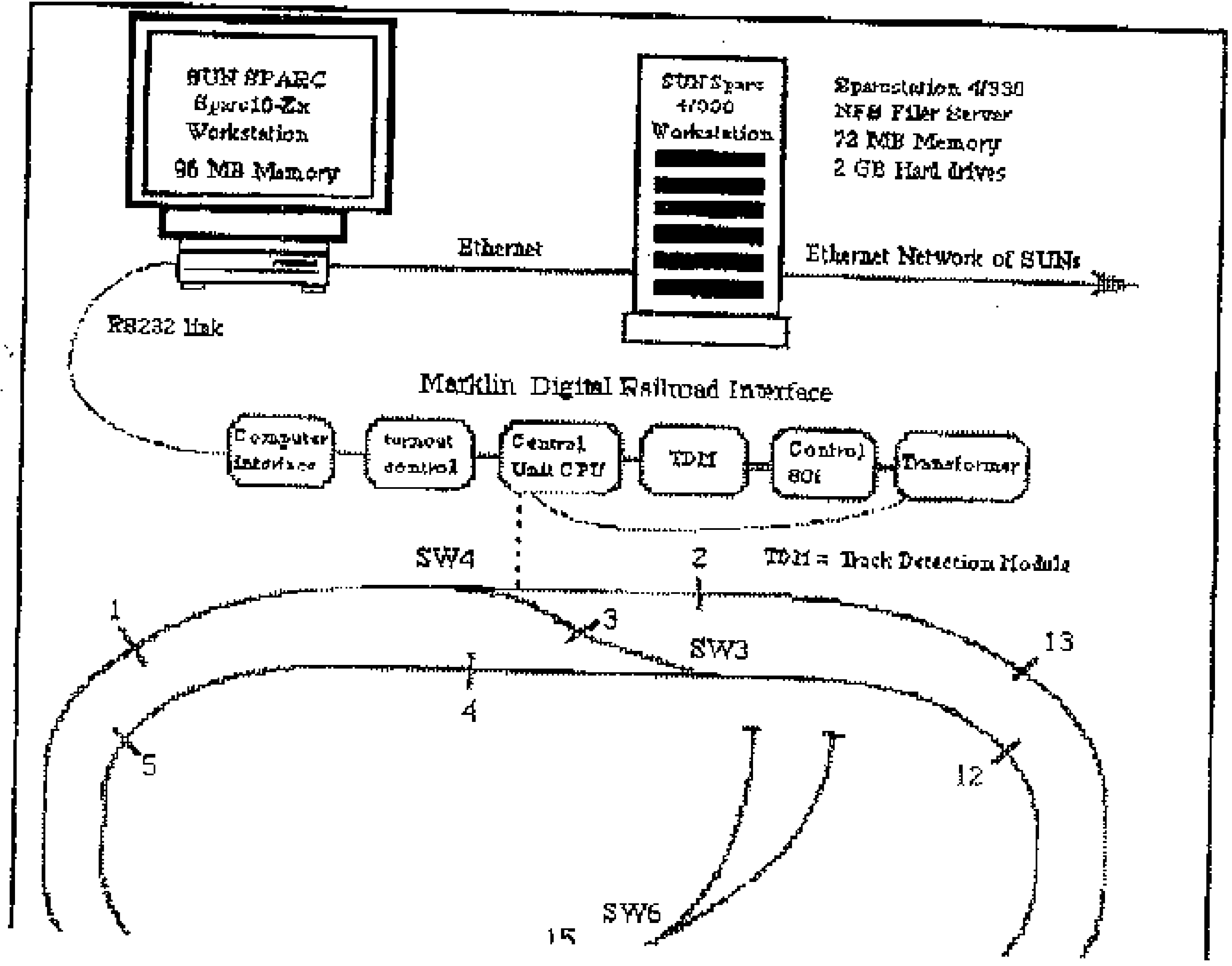
in real-time computing, many do not provide adequate laboratory platforms to exerci to build physical real-time systems[2]. Theoretical modelling and graphics simulati frustrating and spasmodic problems endemic in actual real-time systems. This labora to utilize and exercise their knowledge of mathematics, physics, engineering, compu programming.

II. Equipment, Hardware and Software.

The computer controller is a SUN SPARC workstation connected to a SUN 4/330 file se SUN IPC workstation has 16 MB memory and a 207 MB Hard Disk drive. The Marklin digi used to interface the SUN computer to the track as depicted in Figure 2. The Markli interconnected components: a Central Unit, Computer Interface, Keyboard Turnout Con Module (TDM), Control 80f, and a Transformer. All Marklin modules or components plu architecture between components.

The Central Unit is the CPU of the Marklin system. The Central Unit receives comman that control turnouts and locomotives[3]. The Central Unit overlays each command on sending a signal to the track where it is received by the specific decoder for whic C82 decoder chip in each locomotive or the K87 turnout decoder for switch tracks). module (TDM) is an encoder which translates the incoming signals from the reed cont that the digital system can then use. The Control 80f module is simply a manual cc and direction of any digital locomotive. The K87 Digital Turnout control module can turnouts. Multiple K87's can be connected in series. The K87 will respond to track Marklin Keyboard component or the Computer Interface module.

Figure 1. Photo of Digital Railroad System.



KATZER DECL. EXHIBIT 2

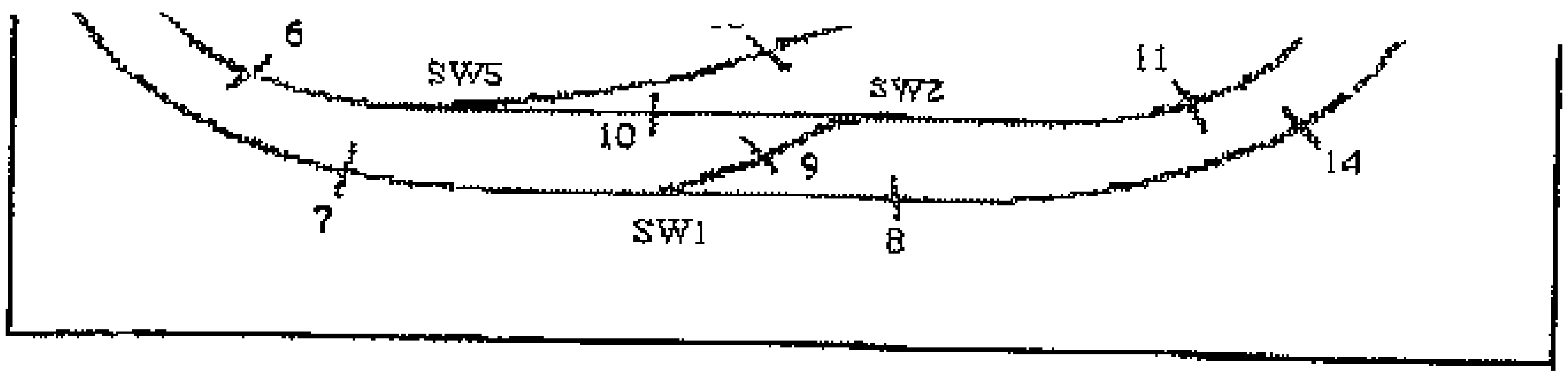


Figure 2. Hardware and Track Layout.
Reed contacts are numbered 1 through 15.
Digital turnouts are numbered SW1 through SW6.

The Computer Interface module is the link between the SUN IPC workstation and the M
system. Using an RS232 9600 baud serial interface, all the functions of the Control
Turnout module can be sent as commands from the computer to the interface module. I
command can be sent to the interface to query the TDM information which specifies w
tripped. In all, up to 80 locomotives, 256 turnout switches, and 496 reed sensors c
the computer interface.
The software is written entirely in 'C'. The 'C' language was chosen as the real-
reasons which are outlined in [4] and [5]. The SUN Developer's Guide was used to ge
Graphic Users Interface (GUI).

III. Device Driver Interface to Marklin System

A device driver was written in 'C' (TRAIN.C) containing the low level commands frc
Marklin Computer Interface hardware via RS232. Functions such TRAIN$START() and TRA
written to initialize and shut down the Marklin system. The function TRAIN$SPEED(tr
the addressable speed command, thus each train could be separately controlled. TRAI
stopped the train train-number, but not the other trains running. TRAIN$SWITCH(swit
straight) would switch the digital turnout to either it's straight or curved positi
reversed the train.
The function call TRAIN$GET-TDM(&tdm1, &tdm2) returns the two bytes sent by the Max
Module. The first byte, tdm1, contains the sensor information for the first 8 sensc
sensors 9 through 16. A magnet on the train will trip the reed sensor when it cross
been tripped. The device latches the bit until a computer command read, which reset
It is interesting to note that a slow train could trip the reed sensor twice. Thus
when the sensor is tripped, read by a computer read command (inquiry), reset to 0,
before the train has completely bypassed the sensor. This is taken care of in the s
reed sensor data.

\Buuns

IV. Software Controller / Concurrent Tasking.

The real-time software controller consists of three separately executing concurrent
(2) a Scheduler and Collision Avoidance task, and (3) a Graphical User Interface (G
prefers to call tasks - processes, in this paper the terms will be used interchange
Graphical User Interface (GUI) task which allows the user to manually control the c
SUN workstation (Figure 4). This task allows the user to: stop, reverse, and chan
address). Also, the user can switch any of the computer connected turnouts on the l
the control mode and unrestricted mode. In control mode the user's requests are sen
the Scheduler Task to determine the viability of the request. Thus, the user is not
would cause a crash. if so, the request is blocked. In unrestricted mode, the user'
Marklin digital system without collision avoidance checks and therefore could cause
The parent process spawns two child processes, the Scan task and the Scheduler task
the two children via a pipe called Control-Pipe. Both children have the ability to
separate conditions. The Scan task will read the control pipe if the GUI task is i
its user commands directly to the Scan task. The Scheduler task will read the con
control mode, in this way user commands initiated from the GUI task will be sent tc
viability then, if viable, sent via the Command-Pipe to the Scan task.
The Scan task has two jobs and continuously loops executing both jobs once for each
job is to collect and decode the current reed

KATZER DECL. EXHIBIT 2

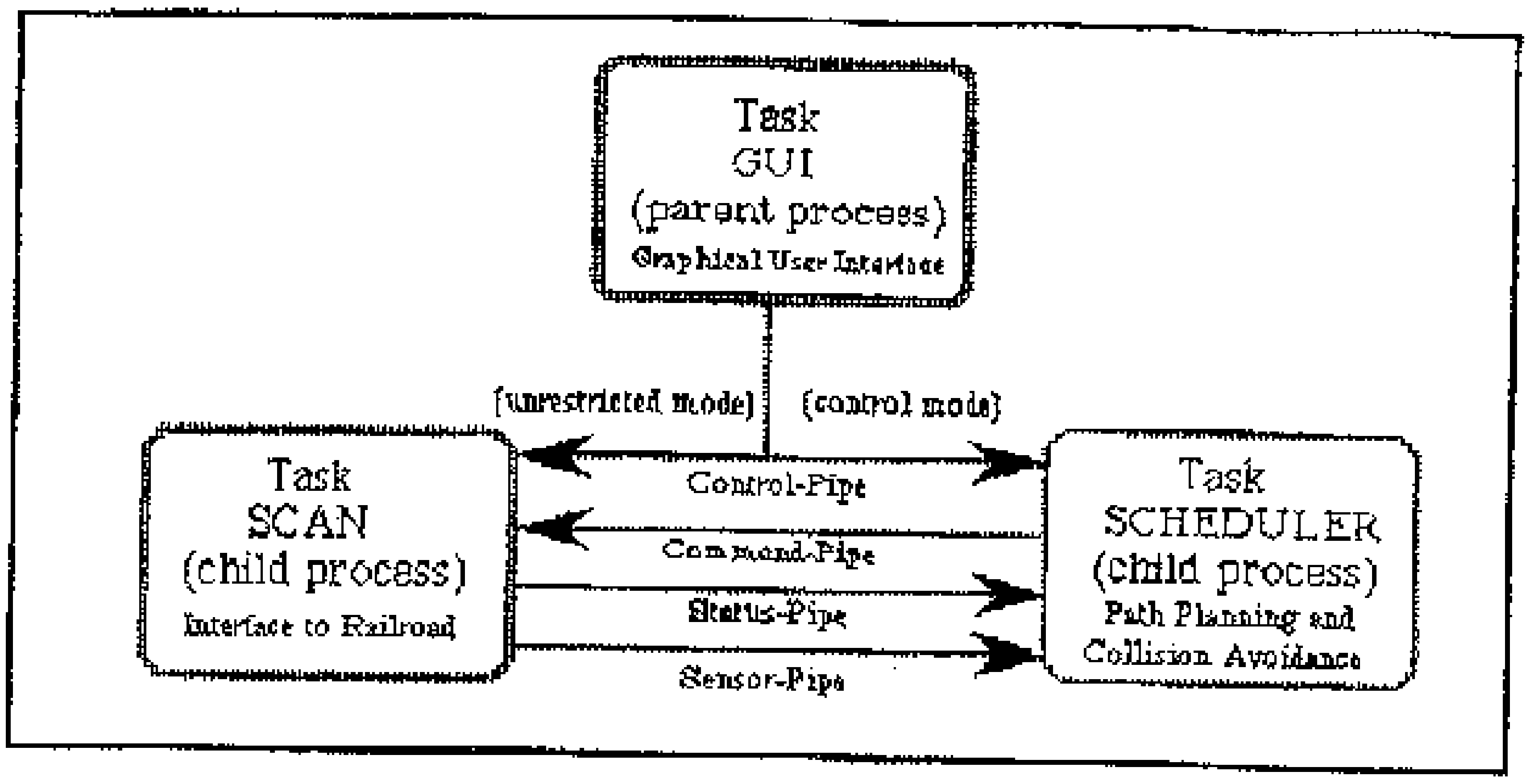


Figure 3. Task Map and Interprocess Communication Piping.

Figure 4. Graphic User Interface (GUI) for Manual Control of the System.
contact sensor information through the Track Detection
Module (TDM). The decoded information is then sent to the Scheduler task for proces
pipe. The Scan task performs this by calling the device driver function TRAIN$GET-T
returns the two bytes sent by the Marklin Track Detection Module. The first byte, t
information for the first 8 sensors on the track. Tdm2 contains sensors 9 through 1
returned in tdm1 and tdm2 is accomplished by combining these two bytes into one wor
comparing to 0x8000 in a for loop. A bit is on if the sensor has been tripped. If t
sensors tripped, at least one of the trains has crossed more than two sensors since
the process is executing too slowly to monitor the traine properly or that some har
exception condition arises, the task immediately issues a TRAIN$HALT() command. Thi
system.
The second job of the Scan task is to relay commands to the Marklin Digital Interfa
only task which accesses the RS232 port connecting the SUN SparcStation to the Mark
Scan task will either accept commands from the Graphical Users Interface task (par
Scheduler task. If the GUI task is in control mode, then the Scan task will receive
which gets its commands from the GUI task (user initiated). If the GUI task is in u
will send the user initiated commands directly to the Scan task without any collisi
commands are sent through the pipe interprocess communication facility in Unix as s
contains command code and the lower nibble contains additional information. when re
train speed adjustment for example.
The Scheduler task is responsible for all control of the system. This task intercep
and determines if current conditions on the train layout will allow the command to
causing a collision or derailment). If so, the command is relayed to the Scan task
Pipe, otherwise the command is blocked from the Marklin system. The Scheduler task
information for each train such as: location, speed, direction, and current zone or
tripped, the sensor value is used to index a lookup table which contains the previc
track layout. In this manner it is possible to monitor the trains without addressab
reed contact will signal the fact that a train (a magnet) has crossed the track. Ho
which train crossed, just that some train (with a magnet) has crossed. Thus, trippi
event. Ambiguity can arise due the fact that tripping a contract is not an address
figures out which train it probabalistically is given the monitoring information it
For example, suppose the current sensor read is 8 and the direction is 0. The previ
is compared to the location of each train in the data structure. If a match is foun
the location field for that train. If no match is found the system issues a TRAIN$H
the system shuts down. In this manner the Scheduler always knows where each train i
allowed to lag behind.
The Scheduler task contains the code to detect collisions. When one train approache
Scheduler either issues a slow down command or a TRAIN$STOP(train-address) to the t
how imminent the collision is. The controller does not want to stop a train unless
imminent situations the Scheduler may issue both a train slow down command to the t

KATZER DECL. EXHIBIT 2

speed up to the front locomotive. Upon each train arriving at a switch, the Schedu
switch, and if so, issues the command to the Marklin system.

V. Conclusion.

This paper has described the work-in-progress of a real-time software controller fo
control software does accomplish its objective of moving the digital locomotives a
according to the scheduling algorithm without collision. Using the Unix real-time p
to perform interprocess communication among three concurrently executing tasks, thi
control of multiple digital locomotives each running on the same track layout while
computerized scheduling system to "run" the trains. The control software continuous
to keep track of each train's location and direction, and is constantly performing
The project was initiated to provide an interesting, experimental platform for the
systems for undergraduates in Computer Science and Computer Engineering. While many
Real-Time Systems acquaint students with the fundamental topics in real-time comput
adequate laboratory platforms to exercise the software skills necessary to build ac
modelling and graphics simulations simply do not manifest the frustrating and spasm
physical real-time systems. This laboratory platform requires students to utilize c
mathematics, physics, engineering, computer science, and real-time programming. A v
system running is available from the authors.

Acknowledgement

This work was partially funded by the National Science Foundation under Grant No. U
Millersville University and the Faculty Grants program of Millersville University.
Dr. Joseph Meier for his expertise on model railroading and for helping us get star
undergraduate research students: Bruce Walters, Chris Coble, Jason Eaby, and David
trade mark of SUN Microsystems, Inc., Unix is a trade mark of AT&T Bell Laborato

References

[1] Webster, Roger and Paul Ross, "A Workstation Laboratory to Improve Undergraduat
National Science Foundation Instrumentation and Laboratory Improvement Program, Gra
University, Millersville, Pennsylvania, USA.

[2] John W. McCormick, "A Model Railroad for Ada and Software Engineering", Commu
1992, Vol. 35, No. 11, pp. 68-70.
P

[3] Catherall, Thomas, "2-Rail Digital DC", Marklin Digital SIG Newsletter, Vol.
1990, pps 1-8.

[4] Ripps, David L. An Implementation Guide to Real-Time Programming, Englewood C
Yourdan Press, 1982, pps. 23-44.

[5] LaPlante Phillip, A.. The Design and Analysis of Real-Time Systems, IEEE Compu
pps. 63-85.

[6] Catherall, Thomas, "Sending Data From the Train to the Digital Component", Ma
New Berlin, Wisconsin, May 1990, pps 1-10.

KATZER DECL. EXHIBIT 2

A Laboratory Platform to Control a Digital Model Railroad Over the Web Using Java *

Roger W. Webster, Ph.D.	Mary A. Klaus
webster@cs.millersv.edu	maklaus@cs.millersv.edu
Timothy A. Bish	Mike Fiorill
tabish@cs.millersv.edu	fiorill@alpha.millersv.edu

Department of Computer Science
Millersville University
Millersville, PA USA 17551

Abstract

This paper describes the work-in-progress of a client-server system to control a digital model railroad over the World Wide Web Using Java. The software engineering objective of this real-time system is to maintain control of multiple digital locomotives each running on the same track layout while at the same time allowing users, anywhere in the world, to manually control the operation of the trains using a java applet running in a web browser. A video camera is connected to the web server showing the users a video stream of the actual physical train system. The java client allows the user to: stop, reverse, and change the speed of any train (by address). Also, the user can switch any of the computer connected turnouts on the layout. The control software (java server) constantly monitors reed contact sensors to keep track of each train's location and direction, and is continuously performing collision avoidance testing. Each digital locomotive and digital turnout switch responds to computer commands that are sent to its address. The computer system, an Intel Pentium running Windows NT®, runs its own web server at http://javatrains.millersv.edu/. This laboratory platform requires students to utilize and exercise their knowledge of mathematics, physics, engineering, real-time programming and computer science.

Introduction

In this railroad layout there are 4 digital turnout switches, two digital locomotives, and fifteen reed contact sensors to manage and control (see Figure 1). The fifteen reed contact sensors are placed in appropriate locations around the track (Figure 2). Magnets are attached to each locomotive which trip reed contact switches which are implanted in the track. This configuration provides an interesting, experimental platform for the study of controlling a real-time system using a java client-server architecture, for undergraduates in Computer Science and Computer Engineering. This laboratory platform requires students to utilize and exercise their knowledge of mathematics, physics, engineering, computer science, and real-time programming. A physical model railroad was used because theoretical modeling and graphics simulations do not always manifest the frustrating and spasmodic problems endemic in actual real-time systems.

*This project was funded, in part, by the National Science Foundation under grant numbers DUE-9350841 and DUE-9651237, and by the Faculty Grants Committee of Millersville University.

Hardware and Equipment

KATZER DECL. EXHIBIT 2

The java server and webserver are run on an Intel Pentium computer running Windows NT® with 32 MB memory and a 1 GB Hard Disk drive. The Marklin® digital railroad system is used to interface the computer to the track as depicted in Figure 2. The Marklin® system is comprised of six interconnected components: a Central Unit, Computer Interface, Keyboard Turnout Control, Track Detection Module (TDM), Control 80f, and a Transformer. All Marklin® modules or components plug together to form a bus architecture between components. The Central Unit is the CPU of the Marklin® system. The Central Unit receives commands from the other modules that control turnouts and locomotives. The locomotives are digitally encoded with a chipset that is addressable, therefore messy block wiring to turn the power on and off is unnecessary. The Central Unit overlays each command on the electric current thereby sending a signal to the track where it is received by the specific decoder for which it is addressed (for example, the C82 decoder chip in each locomotive or the K87 turnout decoder for switch tracks). The S88 Track Detection module (TDM) is an encoder which translates the incoming signals from the reed contact sensors into a data format that the digital system can then use. The Control 80f module is simply a manual control knob for setting the speed and direction of any digital locomotive. The K87 Digital Turnout control module can digitally switch up to four turnouts. Multiple K87's can be connected in series. The K87 will respond to track switch commands from either the Marklin® Keyboard component or the Computer Interface module.

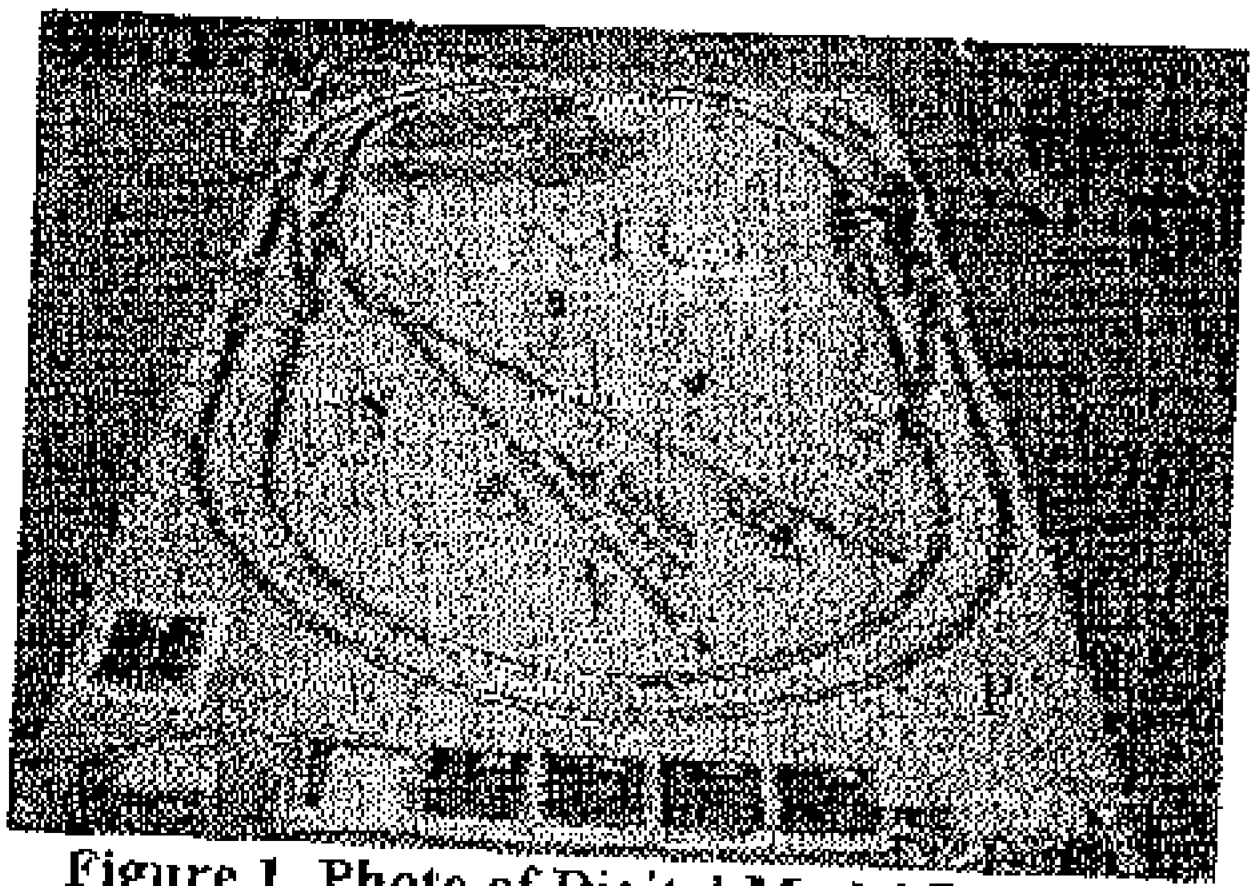


Figure 1. Photo of Digital Model Railroad

The Marklin® Computer Interface module is the link between the computer and the Marklin® Digital HO gauge system. Using an RS232 serial interface, all the functions of the Control 80f and the Keyboard Digital Turnout module can be sent as commands from the computer to the interface module. In addition, a computer command can be sent to the interface to query the TDM information which specifies which reed contacts have been tripped. In all, up to 80 locomotives, 256 turnout switches, and 496 reed sensors can be controlled or monitored with the computer interface.

Interface to Marklin® Digital Railroad System

The java server sends the low level commands from the computer to the Marklin® Computer Interface hardware via RS232. Methods such and TRAINHALT() were written to initialize and shut down the Marklin® system. The method TRAINSPEED(train-number, speed) issued the addressable speed command, thus each train could be separately controlled. TRAINSTOP(train-number) stopped the train train-number, but not the other trains running. TRAINSWITCH(switchnumber, curved-or- straight) would switch the digital turnout to either it's straight or curved position. TRAINREVERSE(train-number) reversed the train. The function call TRAINGET-TDM(tdm1, tdm2) returns the two bytes sent by the Marklin® Track Detection Module. The first byte, tdm1, contains the sensor information for the

KATZER DECL. EXHIBIT 2

first 8 sensors on the track. Tdm2 contains sensors 9 through 16. A magnet on the train will trip the reed sensor when it crosses. A bit is on if the sensor has been tripped. The device latches the bit until a computer command read, which resets it to zero. It is interesting to note that a slow train could trip the reed sensor twice. Thus a double hit occurs. This happens when the sensor is tripped, read by a computer read command (inquiry), reset to 0, then read again by the software before the train has completely bypassed the sensor. This is taken care of in the software by masking off the previous reed sensor data.

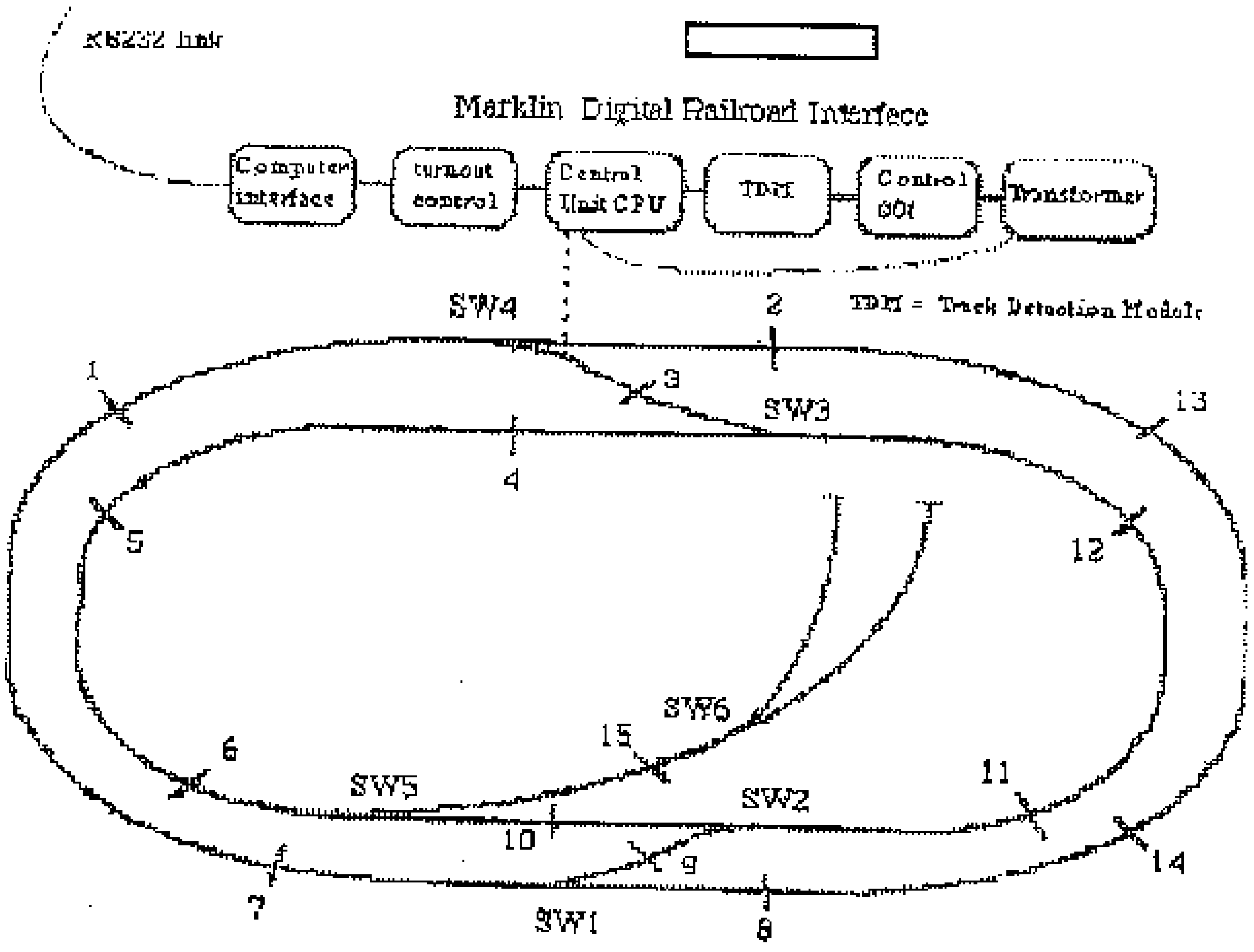


Figure 2. Track Layout. Reed contacts are numbered 1 through 15. Digital turnouts are numbered SW1 through SW6.

Java Client - The User Interface

The java client (see figure 3) allows the user to manually control the operation of the trains from anywhere in the world. This java applet allows the user to: stop, reverse, and change the speed of any train (by address). Also, the user can switch any of the computer connected turnouts on the layout. The java client sends commands to the server to determine the viability of the request. Thus, the user is not permitted to make a change that would cause a crash. If so, the request is denied by the server.

KATZER DECL. EXHIBIT 2

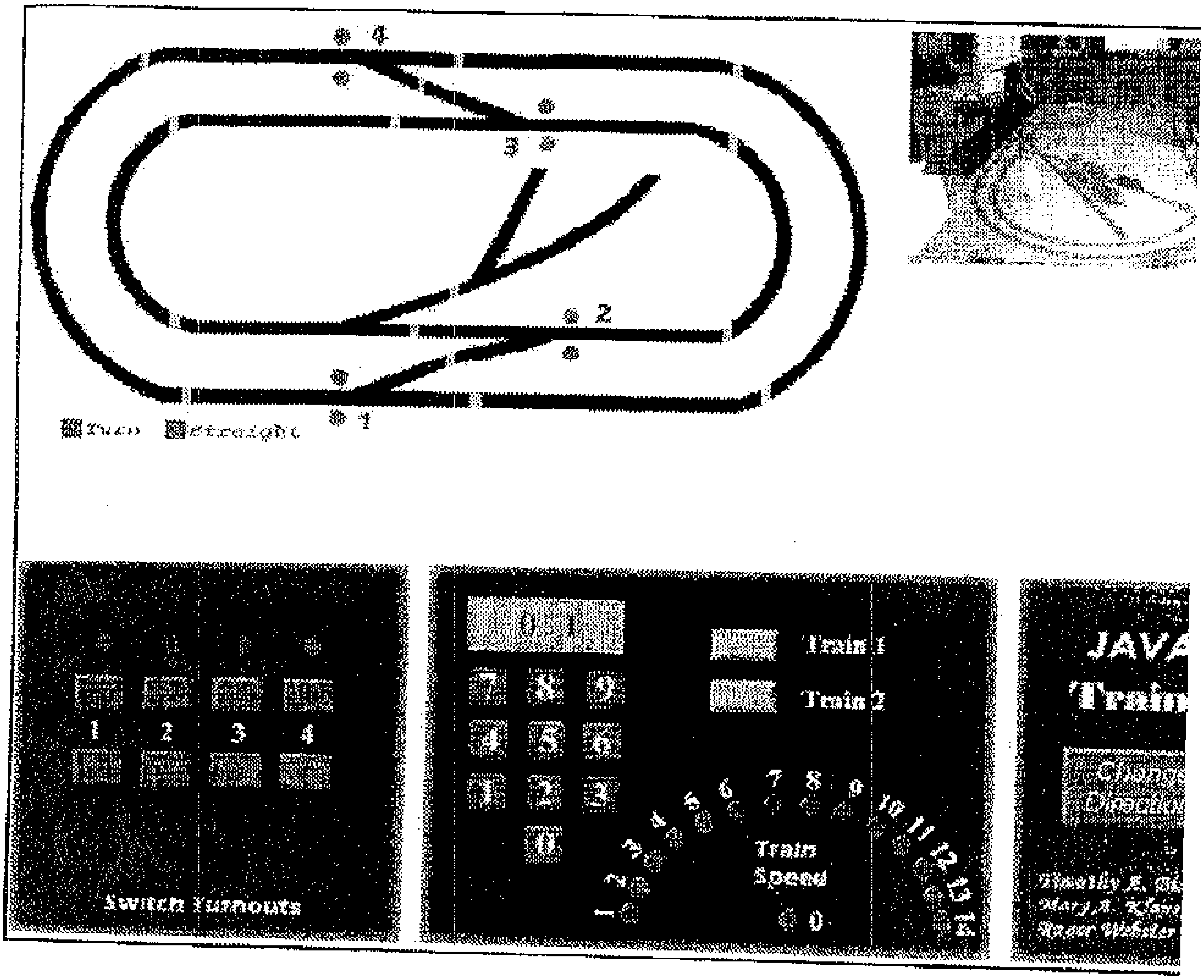


Figure 3. Java Client - The User Interface.

Java Server - The Software Controller

The java server is actually three separate tasks all continuously looping and executing their jobs once for each pass through their loop. The first task is the server to the client. This process simply takes commands from the client and passes them on to the next task, the AI. A timeout is set up to notify the client that something has gone wrong and ask him to restart if the tasks take too long to respond. The simplicity of this task reduces its chance of failure so that the user can be kept informed if other problems occur.

Tuesday, May 09, 2006 (2).max

KATZER DECL. EXHIBIT 2

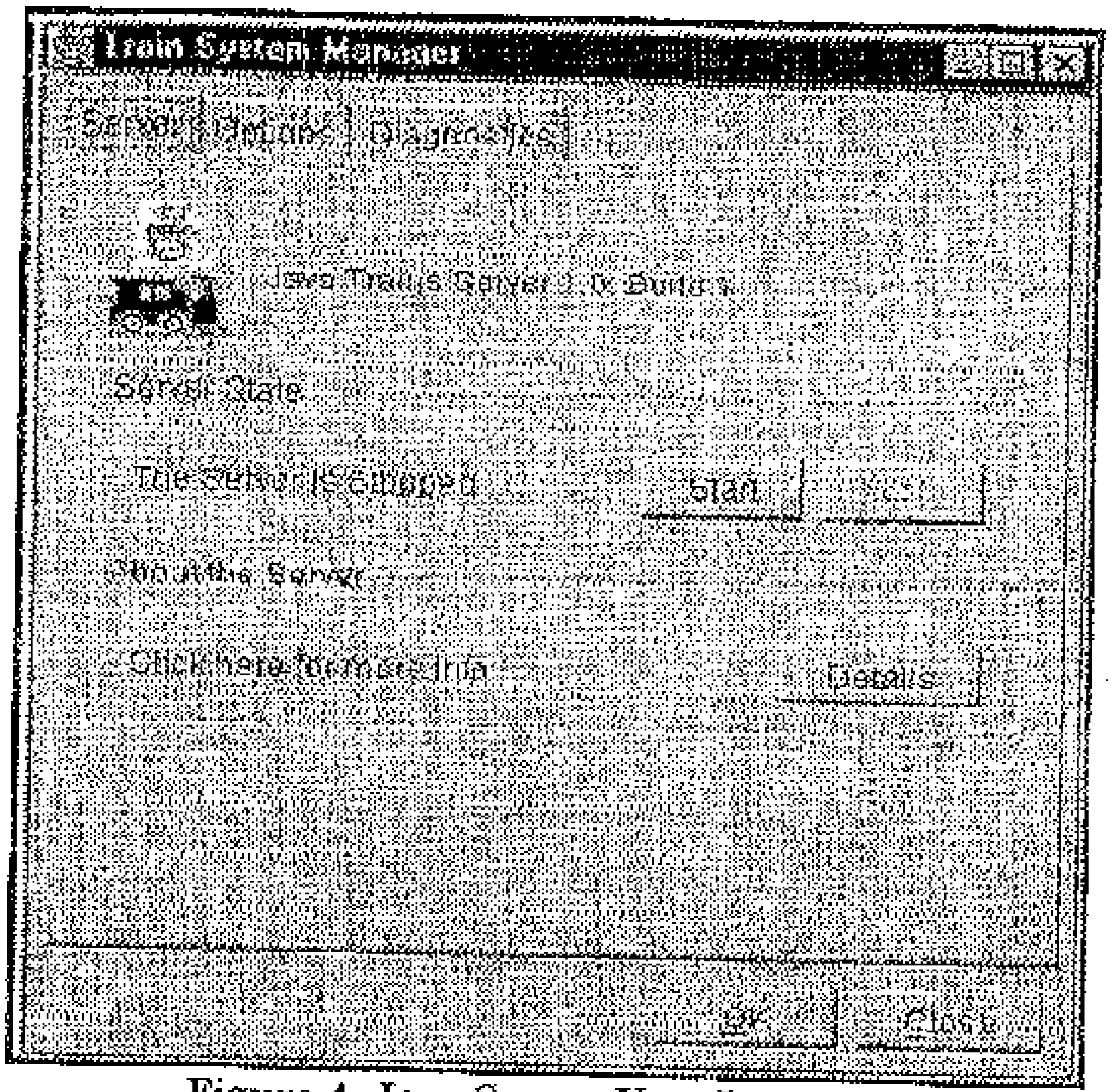


Figure 4. Java Server User Interface.

The second task, the AI receives TDM data from the scan task and uses this to keep track of the positions of each of the trains. If a command sent by the user would result in a collision it modifies or ignores the command and sends this information back to the client so the interface can be updated. The AI contains the code to detect collisions. When one train approaches another train too closely, the AI either issues a slow down command or a stop(train-address) to the train behind, depending on how imminent the collision is. The controller does not want to stop a train unless it is imperative to do so. In imminent situations the AI may issue both a train slow down command to the rear locomotive, and a train speed up to the front locomotive. If the user issues a command the flip a switch, the AI determines if it is safe to switch, and if so, issues the command to the Marklin® system.

The scan task is the one that actually talks to the trains. It receives commands from the AI and sends these out to the trains. All commands are sent as one or two bytes. The first byte contains the command code and the second (when appropriate) contains additional information such as train speed. When there are no commands coming in, the scan task continuously asks for TDM data from the trains. This task also makes sure that only one command is sent between successive TDM calls. The scan task gets TDM data by calling the method getTDM() which returns the two bytes sent by the Marklin® Track Detection Module. The first byte contains the sensor information for the first 8 sensors on the track. The second byte contains sensors 9 through 16. The decoding of the sensor data returned by getTDM is accomplished by left shifting the first byte and combining these two bytes into one word. This data is then sent on to the AI.

The AI knows the current direction (forward or backward) of each train, its previous position (which sensor it last tripped) and the state (straight or curved) or each switch. However, the contact does not know which train crossed, just that some train (with a magnet) has crossed. Thus, tripping a contact is not an addressable event Ambiguity can arise due the fact that tripping a contact is not an addressable event. The AI task figures out which train it probably is given the monitoring information it is

KATZER DECL. EXHIBIT 2

maintaining. Using this information, it translates the TDM data into a new position for each train by looking up information about possible next positions for each train in an array. For example, if a train was previously at sensor 11 and all switches were straight, it shouldn't be at sensor 9 the next time. A bit is on if the sensor has been tripped. If the function returns more than two sensors tripped, at least one of the trains has crossed more than one sensor since the last update or some hardware

All commands are sent as one byte. The upper nibble contains command code and the lower nibble contains additional information, when required, such as in the case of train speed adjustment for example. The server task is responsible for all control of the system. This task accepts all user commands from the java client and determines if current conditions on the train layout will allow the command to be executed safely (without causing a collision or derailment). If so, the command is executed otherwise the command is blocked from the Marklin® system. The server task keeps track of vital information for each train such as: location, speed, direction, and current zone or sector.

Each time a sensor is tripped, the sensor value is used to index a lookup table which contains the previous value for each sensor on the track layout. In this manner it is possible to monitor the trains without addressable track detection information. The reed contact will signal the fact that a train (a magnet) has crossed the track. However, the contact does not know which train crossed, just that some train (with a magnet) has crossed. Thus, tripping a contact is not an addressable event. Ambiguity can arise due the fact that tripping a contract is not an addressable event. The java server control software figures out which train it probably is given the monitoring information it is maintaining.

For example, suppose the current sensor read is 8 and the direction is 0. The previous sensor would be 14. This value is compared to the location of each train in the data structure. If a match is found the current sensor value is stored in the location field for that train. If no match is found the system issues a TRAINHALT indicating a lost train, and the server shuts down. In this manner the server always knows where each train is at any time and is never allowed to lag behind.

The java server contains the code to detect collisions. When one train approaches another train too closely, the server either issues a slow down command or a TRAINSTOP(train-address) to the train behind depending on how imminent the collision is. The controller does not want to stop a train unless it is imperative to do so. In imminent situations the server may issue both a train slow down command to the rear locomotive, and a train speed up to the front locomotive. Upon each train arriving at a switch, the server determines if it is safe to switch, and if so, issues the command to the Marklin® system.

Conclusion

This paper has described the work-in-progress of a java client-server controller for a digital model railroad. The control software does accomplish its objective of maintaining control of multiple digital locomotives each running on the same track layout while at the same time allowing users around the world to manually control the operation of the trains using a java applet running in a web browser. A video camera is connected to the web server showing the users a video stream of the train system. The java client allows the user to: stop, reverse, and change the speed of any train (by address). Also, the user can switch any of the computer connected turnouts on the layout. The control software constantly monitors reed contact sensors to keep track of each train's location and direction, and is continuously performing collision avoidance testing.

The project was initiated to provide an interesting, experimental platform for the study of controlling a real-time system over the world wide web with a java client-server architecture. This laboratory platform requires students to utilize and exercise their knowledge of mathematics, physics, engineering,

KATZER DECL. EXHIBIT 2

real-time programming and computer science. Further information and source code can be found on our web site at http://cs.millersv.edu/javatrains/.

Acknowledgements

This project was funded, in part, by the National Science Foundation under grant numbers DUE-9350841 and DUE-9651237, and by the Faculty Grants Committee of Millersville University. Many thanks go to Mrs. Bonnie Work, for administrative assistance. Special thanks go to Robert Sauders setting up the DNS entry javatrains.millersv.edu.

References

- Catherall, Thomas, "2-Rail Digital DC", Marklin® Digital SIG Newsletter, Vol. 2, No. 1, New Berlin, Wisconsin, January 1990, pps 1-8.
- Catherall, Thomas, "Sending Data From the Train to the Digital Component", Marklin® Digital SIG Newsletter, Vol. 2 No. 3, New Berlin, Wisconsin, May 1990, pps 1-10.
- Cassidy, Luke, "Industrial Strength Java", New Riders Press, Indianapolis, IN., 1997.
- Flanagan, David, "Java 1.1 in a Nutshell", Second Edition, O'Reilly Press, Cambridge, Massachusetts, 1997.
- Flanagan, David, "Java 1.1 Examples", O'Reilly Press, Cambridge, Massachusetts, 1997.
- Geary, David M., "Graphic Java 1.1 - Mastering the AWT", Second Edition, Sun Microsystems Press, Mountain View, CA, 1997.
- McCormick, John W., "A Model Railroad for Ada and Software Engineering", Communications of the ACM, November 1992, Vol. 35, No. 11, pp. 68-70.

Department of Computer Science
Millersville University

Java Trains Web Site

KATZER DECL. EXHIBIT 2

Katzer Declaration: Exhibit 3

Katzer Declaration: Exhibit 3

Matt Katzer

From: Bob Jacobsen [Bob_Jacobsen@lbl.gov]
Sent: Monday, October 22, 2001 3:50 PM
To: loconet_hackers@yahoogroups.com
Subject: Re: [loconet_info] Standard API for talking to loconet.

At 2:59 PM -0700 10/22/01, Christopher Coley wrote:
>Is there a standard API for talking to Loconet, or any other control
>bus for that mater. It must work with Loconet. Or is everybody doing
>there own thing???
>
>To make software easier for us all, If there is not a standard cross
>platform API I suggest we develop one.

"The nice thing about standards is that there are so many of them to
choose from." - Andrew Tanenbaum

There are several efforts to do similar things. There's a list with links at the bottom of http://jmri.sourceforge.net/Technical/index.html . You can also find them from Don Crano's web ring.

One effort in that direction is JMRI. Although by no means complete, it provides various tools for talking to LocoNet and NCE equipment.
Its got low-level support for handling the LocoNet protocol via MS100 or LocoBuffer, and higher level stuff to let you code at the level of "write CV" or "set turnout". Its in java, which has pluses and minuses. For more info, see http://jmri.sourceforge.net

Bob
--

Bob Jacobsen (Bob_Jacobsen@lbl.gov, 510-486-7355, fax 510-495-2957)

To unsubscribe from this group, send an email to:
loconet_hackers-unsubscribe@yahoogroups.com

Your use of Yahoo! Groups is subject to http://docs.yahoo.com/info/terms/

KATZER DECL. EXHIBIT 3

Matt Katzer

From:	Bob Jacobsen [Bob_Jacobsen@lbl.gov]
Sent:	Tuesday, November 13, 2001 9:43 AM
To:	loconet_hackers@yahoogroups.com
Subject:	Re: LocoNet bridge (was Re: [loconet_info] locobuffer)

At 1:08 PM +0000 11/13/01, Juergen Hauschild wrote:
>Here in Germany we have a lot of initiatives of citizens who want the
>government to forbid the DECT as well as the GSM technology because it
>influences our biological communication via the nerve system.
>A lot of experts have done studies and they show up that pulsed
>microwaves which are send out from the DECT and GSM can be very
>dangerous to human live. Lots of effects are described in the
>international literature.

Although its off-topic to some extent, I think a reply to this is appropriate. If you'd like to discuss this more, please contact me off-list.

Juergen's statement is _not_ consistent with the scientific literature.

My background: I was trained as an electrical engineer, and am currently a Physics Professor at the University of California at Berkeley. I've studied the question of biological effects of EM fields at some length.

In 1999, the United States Food & Drug Administration surveyed the literature and found "no scientific evidence" of health hazards from consumer-level exposure to microwaves, including cell phones of all three types in use in the US. That result explicitly included frequencies around 2GHz used by DECT, although DECT was not mentioned by name.

Additional studies were funded and completed to cover certain topics where the evidence was incomplete or inconclusive. Those were again surveyed by the U.S. GAO (an arm of Congress), who concluded in May
2001 that "The consensus of the Food and Drug Administration (FDA), the World Health Organization, and other major health agencies is that the research to date does not show radiofrequency energy emitted from mobile phones has harmful health effects, but there is not yet enough information to conclude that they pose no risk."

Various groups (_not_ scientific organizations!) have interpreted the final phase to mean "the studies haven't shown anything, there's still a large risk". That is an incorrect interpretation. What the studies have shown is that any risks that are present for the vast majority of the public are at least a factor of 3000 smaller than the (mostly unknown) factors that cause the general level of health of the population. What we don't know is whether there is exists one or more small (less than 0.1% of the population) subgroups who may have a sensitivity resulting a small increase in adverse health effects, e.g. cancer. As you can imagine, this is a very hard thing to prove or disprove. Should such populations exist with worst-case sensitivity, the total cancer rate in the US would increase by 0.004%.

If you're interested in learning more, a good place to start is with the GAO report, available online at http://www.gao.gov/new.items/d01545.pdf or via the GAO web site. I can also provide direct references in the scientific literature to specific studies and meta-studies done in US, UK, France, Germany and Sweden. Unfortunately, these are not generally available on the web but should be obtainable in any university library. Please contact me off-list if you're interested.

I encourage people to make their own choices for their own use of technology. That's particularly true for a hobby activity such as DCC. Personally, I've concluded that the risk from cellphones is outweighed by the benefits. I use one, as do other members of my family. I also think that attempts to significantly reduce use of microwave technology are not motivated by scientific evidence, but rather by other political and ideological factors.

KATZER DECL. EXHIBIT 3

Bob
--

Bob Jacobsen (Bob_Jacobsen@lbl.gov, 510-486-7355, fax 510-495-2957)

To unsubscribe from this group, send an email to:
loconet_hackers-unsubscribe@yahoogroups.com

Your use of Yahoo! Groups is subject to http://docs.yahoo.com/info/terms/

KATZER DECL. EXHIBIT 3

Matt Katzer

From: Bob Jacobsen [Bob_Jacobsen@lbl.gov]
Sent: Wednesday, January 23, 2002 7:53 AM
To: loconet_hackers@yahoogroups.com
Subject: [loconet_info] What happens on LocoNet when Empire Builder is asked to read?

I'd like to ask a favor of the group.

I want to make sure that the Decoder Pro program works well with an Empire Builder (EB). In particular, I'd like to have it detect that its working with an Empire Builder so that it can disable the "read"
buttons.

But I don't know how to identify that the command station is an Empire Builder, rather than anything else. Does anybody know how to do that _before_ attempting a read?

As a minimum, I'd like to properly handle how the EB reports that it can't read. To do that, I'd like to know exactly what happens on the LocoNet during a read request to the EB. Could somebody with an Empire Builder attempt to do a programming read with a throttle, and send me a LocoNet trace of the resulting traffic? Any format will be fine.

Finally, would there be anybody on this list in Northern California who's got an Empire Builder and would be willing to let me talk to it for a little while? I'd like to check my code updates to make sure they really work before sending them to some unsuspecting victim^h^h^h^h^h^h volunteer. I'm willing to drive pretty far.

Thanks in advance.

Bob
--

Bob Jacobsen (Bob_Jacobsen@lbl.gov, 510-486-7355, fax 510-495-2957)

KATZER DECL. EXHIBIT 3

Matt Katzer

From:	Bob Jacobsen [Bob_Jacobsen@lbl.gov]
Sent:	Tuesday, August 26, 2003 9:52 AM
To:	digitrax@yahoogroups.com
Subject:	[Digitrax] Re: Best way to match speeds?

At 2:54 PM +0000 8/26/03, Craig wrote:
>It looks like we are breaking new ground here. I didn't get any ideas
>from my posting either. I also emailed Digitrax hoping that they had
>an app. note and didn't get any good suggestions other than use OPS
>mode programming to match on the fly or use a PR1.

If you're willing to use a computer with a LocoNet connection, you might want to try DecoderPro with ops-mode programming. You can open a programmer panel for each of several locomotives, and tweak their speed tables separately while they're running until everything is all matched up.

For more info: http://jmri.sf.net/DecoderPro

(Mandatory disclaimer: I do some work on DecoderPro, but since it's a free program I don't get any money from it)

Bob
--

Bob Jacobsen (Bob_Jacobsen@lbl.gov, 510-486-7355, fax 510-495-2957) Am in Austria this week, email may be slow.

To Unsubscribe From This List: send an email to: digitrax-unsubscribe@yahoogroups.com

Your use of Yahoo! Groups is subject to http://docs.yahoo.com/info/terms/

KATZER DECL. EXHIBIT 3

Matt Katzer

From: Bob Jacobsen [Bob_Jacobsen@lbl.gov]
Sent: Tuesday, January 20, 2004 8:05 PM
To: jmriusers@yahoogroups.com
Subject: Re: [jmriusers] javax.comm for MacOS X 10.3.2

(I tried to send this off-list, but the email address on the Yahoo account isn't correct. Could you please check that and fix it?
Thanks!)

At 8:15 PM +0000 1/20/04, puzzle22000 wrote:

```
>After double klicking the installer I'm asked to give my password.
>Then nothing happens.
>Again double klicking on the installer opens the installer window.
>I decide to install the MacOS X version.
>Start Installation.
>A message box appears:
>    Vise    Error creating file.
>    1008:,-5000 Access denied error
>Klicking ok in the box
>A second box appears:
>    You don't have enoug access permissions for this installation.
>
>I am using a PowerBook 15" , with MacOS X 10.3.2.
>I am the only user of this machine and could install everything else before.
>
>Please give me some hints.
```

I just ran the installer ("Java comm installer 2") on a new 10.3.2 system without encountering that.

It does sound like a permissions problem, so the possibilities include:

1) Something wrong with file permissions. You should try running "repair file permissions" from the disk tool.

2) The account you're using isn't the admin account (as Dave Falkenburg suggested in another email). I'm not sure what the easiest way to check that is. System preferences -> accounts shows you the accounts on the left side, and identifies the "admin" account(s).

Bob

--

Bob Jacobsen (Bob_Jacobsen@lbl.gov, 510-486-7355, fax 510-495-2957)

Yahoo! Groups Links

To visit your group on the web, go to:
http://groups.yahoo.com/group/jmriusers/

To unsubscribe from this group, send an email to:
jmriusers-unsubscribe@yahoogroups.com

Your use of Yahoo! Groups is subject to:
http://docs.yahoo.com/info/terms/

KATZER DECL. EXHIBIT 3

Matt Katzer

From: Bob Jacobsen [Bob_Jacobsen@lbl.gov]
Sent: Wednesday, January 21, 2004 10:44 AM
To: jmriusers@yahoogroups.com
Subject: Re: [jmriusers] Havng problems with preferences

At 7:53 AM +0000 1/21/04, Chuck Conner wrote:
>I have downloaded several versions of Decoderpro and can get the
>program to start. But when I enter my preferences they are not
>remembered and so I cannot use the program. I have used the program on
>an old desktop without problems but have loaded it on an IBM thinkpad
>600 for use at the layout. I have tried Locotools to create throttles
>and they will not accept addresses. Can anyone help? Tell me what I am
>doing wrong.

Please try the most recent 1.3.8a test version. It's supposed to fix a problem with
Windows XP and 2K that sounds a lot like that.

Bob
--

Bob Jacobsen (Bob_Jacobsen@lbl.gov, 510-486-7355, fax 510-495-2957)

KATZER DECL. EXHIBIT 3

Matt Katzer

From: jmriusers@yahoogroups.com on behalf of Bob Jacobsen [Bob_Jacobsen@lbl.gov]
Sent: Wednesday, June 08, 2005 12:49 PM
To: jmriusers@yahoogroups.com
Subject: Re: [jmriusers] Digitrax Cab Signalling

At 5:16 PM +0000 6/8/05, scrhenry wrote:
>Has anyone developed an implementation of JMRI to take advantage of cab
>signialing feature that comes with DT300 and DT400 throttles?

There's some messaging code in JMRI for doing that. It will let you send alpha-numeric or "semaphore" messages to throttles.

I think that with a little work, somebody could bring it up to a nice general capability. But it's not quite there yet.

If you'd like to play with it, I can send you pointers to the classes, and help you get started with a sample script.

Bob
--
Bob_Jacobsen@lbl.gov +1-510-486-7355 fax +1-510-643-8497 AIM, Skype JacobsenRG

Yahoo! Groups Links

<*> To visit your group on the web, go to:
    http://groups.yahoo.com/group/jmriusers/

<*> To unsubscribe from this group, send an email to:
    jmriusers-unsubscribe@yahoogroups.com

<*> Your use of Yahoo! Groups is subject to:
    http://docs.yahoo.com/info/terms/

KATZER DECL. EXHIBIT 3

Matt Katzer

From: jmri-developers-admin@lists.sourceforge.net on behalf of Bob Jacobsen [Bob_Jacobsen@lbl.gov]
Sent: Thursday, April 13, 2006 10:01 AM
To: jmri-developers@lists.sourceforge.net
Subject: Re: Signalling logic (was Re: [Jmri-developers] Status of HEAD of CVS

(Sorry, bad week at the day job)

At 10:25 PM -0400 4/8/06, Dick Bronson wrote:
>
>As I understand your proposal the CTC logic would reside in its own
>"Simple Traffic Control" section under tools. That would be the basic
>structure that you would add. There would then be a whole new section
>added that has a similar structure as the BlockBoss has.

Right. The idea here is to leave BlockBoss (simple or vital signal logic) alone, but control it from outside. That prevents BlockBoss from getting much more complicated than in already is and allows the user to separate configuring the two parts.

>The hook to "Hold" a signal red would be done simply by entering the
>signal to "Hold", and setting it. I missed that concept when reading in
>the comments, and was making the assumption that to hold a signal red
>would require that a special sensor be added to the list of sensors in
>the SSL sections for that purpose. As I understand it now, the SSL
>would not have any different settings in a system with CTC as it would
>have for one without.

In almost all circumstances that's right. The one exception is if we eventually want to have "call-on" signals, but I haven't thought them through entirely yet.

>I'm still thinking in terms of using virtual turnout and virtual signal
>items to get the lever graphics onto the panel. The biggest problem I
>see with that is that the signal states go R-Y-G-R-Y-G....
>as you click on them. If they went R-Y-G-Y-R-Y-G... then the lever
>image could go back and forth nicely rather than jumping from one side
>to the other. I don't think that change would affect anything else
>negatively would it? People don't normally click on signals to change
>them except for testing do they?

I'd prefer to use the "memory" beans for this. They can have an arbitrary set of icons & sequencing. I'll put together an example.

The real question is how do we want them to behave. I _think_ that the best approach would be "click on the left to put the lever to the left", etc, rather than have a sequence. But it makes the click target noticably smaller, particularly if the lever has more than two routes.

>While on the subject of signals, there is a bit of a disconnect in the
>system between the "Change signal icons" window and the "Signal Table".
>The Signal Table allows you to cycle through states that have no
>matching icon selection options. Specifically, "Flashing Red",
>"Flashing Green", and "Dark", are all missing and default to some small
>images with no obvious way to change them graphically.

yes, this needs to be fixed.

>Back to the subject, I further presume that this new section would be
>the place for not only the CTC logic, but also APB logic using "stick"
>circuitry to tumble down all opposing traffic signals in a single track
>area. Also maybe interlocking would fit there. By automated running do
>you mean scripted train operations?

KATZER DECL. EXHIBIT 3

There are really two kinds of tumble down logic. True APB is _Automatic_, operating when a train enters the controlled section.
Most model railroads don't have enough blocks between sidings for this to really work right.

CTC usually had something different, called "reserved direction".
Basically, when a traffic lever was set to allow entry to the single track between OS sections & coded, the signals would clear for that entire section, and the opposing OS signals (entry to the single track from the other end) would be unable to clear. (Route and/or time locking was added on top of this in case the dispatcher changed his mind before a train went through)

(Another way to put this: The lever is really selecting direction, not signals. The vital signalling logic converts the direction into consistent signal aspects for the entire line)

Reserved direction should probably live in this new chunk of code, along with the CTC logic. I'm not sure whether it can always be present, or whether it would have to be controllable. Some people might not like it because it links the operation of the direction & turnout levers at the two ends in a way that can be quite confusing if you haven't worked through why it's happening.

>I will not have any chance to work on this for a few days, because I
>have been asked to take a trip to Dallas with a 40' ship container.
>I am supposed to leave the rig there for another driver to pick up in a
>few weeks, and they will fly me back home, probably on Wednesday.
>Supposedly I have hours of "free" time due to the new CDL regulations,
>but in my experience it is still; eat, drive, eat, drive, eat, drive,
>sleep, eat, etc. till you finish the trip.

Hope you had a great trip!

>
>I have started the process of programming the robot to build Tower
>Controllers, so maybe I can start filling some back orders on them as
>well. Also I have committed to having my own layout's signals running
>for an open house at the end of May. Two weeks of the time between now
>and then is committed to other travel, so I guess I had better not take
>many naps.

Let me know if I can help with this.

Bob

--
Bob_Jacobsen@lbl.gov +1-510-486-7355 fax +1-510-643-8497 AIM, Skype JacobsenRG

KATZER DECL. EXHIBIT 3

Katzer Declaration: Exhibit 4

Katzer Declaration: Exhibit 4

Matt Katzer

From:	Bob Jacobsen [Bob_Jacobsen@lbl.gov]
Sent:	Tuesday, November 13, 2001 9:43 AM
To:	loconet_hackers@yahoogroups.com
Subject:	Re: LocoNet bridge (was Re: [loconet_info] locobuffer)

At 1:08 PM +0000 11/13/01, Juergen Hauschild wrote:
>Here in Germany we have a lot of initiatives of citizens who want the
>government to forbid the DECT as well as the GSM technology because it
>influences our biological communication via the nerve system.
>A lot of experts have done studies and they show up that pulsed
>microwaves which are send out from the DECT and GSM can be very
>dangerous to human live. Lots of effects are described in the
>international literature.

Although its off-topic to some extent, I think a reply to this is appropriate. If you'd like to discuss this more, please contact me off-list.

Juergen's statement is _not_ consistent with the scientific literature.

My background: I was trained as an electrical engineer, and am currently a Physics Professor at the University of California at Berkeley. I've studied the question of biological effects of EM fields at some length.

In 1999, the United States Food & Drug Administration surveyed the literature and found "no scientific evidence" of health hazards from consumer-level exposure to microwaves, including cell phones of all three types in use in the US. That result explicitly included frequencies around 2GHz used by DECT, although DECT was not mentioned by name.

Additional studies were funded and completed to cover certain topics where the evidence was incomplete or inconclusive. Those were again surveyed by the U.S. GAO (an arm of Congress), who concluded in May
2001 that "The consensus of the Food and Drug Administration (FDA), the World Health Organization, and other major health agencies is that the research to date does not show radiofrequency energy emitted from mobile phones has harmful health effects, but there is not yet enough information to conclude that they pose no risk."

Various groups (_not_ scientific organizations!) have interpreted the final phase to mean "the studies haven't shown anything, there's still a large risk". That is an incorrect interpretation. What the studies have shown is that any risks that are present for the vast majority of the public are at least a factor of 3000 smaller than the (mostly unknown) factors that cause the general level of health of the population. What we don't know is whether there is exists one or more small (less than 0.1% of the population) subgroups who may have a sensitivity resulting a small increase in adverse health effects, e.g. cancer. As you can imagine, this is a very hard thing to prove or disprove. Should such populations exist with worst-case sensitivity, the total cancer rate in the US would increase by 0.004%.

If you're interested in learning more, a good place to start is with the GAO report, available online at http://www.gao.gov/new.items/d01545.pdf or via the GAO web site. I can also provide direct references in the scientific literature to specific studies and meta-studies done in US, UK, France, Germany and Sweden. Unfortunately, these are not generally available on the web but should be obtainable in any university library. Please contact me off-list if you're interested.

I encourage people to make their own choices for their own use of technology. That's particularly true for a hobby activity such as DCC. Personally, I've concluded that the risk from cellphones is outweighed by the benefits. I use one, as do other members of my family. I also think that attempts to significantly reduce use of microwave technology are not motivated by scientific evidence, but rather by other political and ideological factors.

KATZER DECL. EXHIBIT 4

Bob
--

Bob Jacobsen (Bob_Jacobsen@lbl.gov, 510-486-7355, fax 510-495-2957)

KATZER DECL. EXHIBIT 4