Final Research Report
Research Project T9903, Task 43
Busview

BUSVIEW:
AN APTS PRECURSOR AND A DEPLOYED APPLET

by

D.J. Dailey, S. Maclean, I. Pao
ITS Research Program
College of Engineering, Box 352500
University of Washington
Seattle, Washington 98195-2500

Washington State Transportation Center (TRAC)
University of Washington, Box 354802
University District Building
1107 NE 45th Street, Suite 535
Seattle, Washington 98105-4631

Washington State Department of Transportation
Technical Monitor
Morgan Balogh

Prepared for

Washington State Transportation Commission
Washington State Department of Transportation
Olympia, Washington 98504-7370

And in cooperation with
U.S. Department of Transportation
Federal Highway Administration

June 2000

TECHNICAL REPORT STANDARD TITLE PAGE

1. REPORT NO. WA-RD 467.1	2. GOVERNMENT ACCESSION NO.	3. RECIPIENT'S CATALOG NO.
4. TITLE AND SUBTITLE Busview: an APTS Precursor and a Deployed Applet		5. REPORT DATE June 2000 6. PERFORMING ORGANIZATION CODE
7. AUTHOR(S) D.J. Dailey, S. Maclean, I. Pao		8. PERFORMING ORGANIZATION REPORT NO.
9. PERFORMING ORGANIZATION NAME AND ADDRESS Washington State Transportation Center (TRAC) University of Washington, Bx 354802 University District Building; 1107 NE 45th Street, Suite 535 Seattle, Washington 98105-4631		10. WORK UNIT NO. 11. CONTRACT OR GRANT NO. Agreement T9903, Task 43
12. SPONSORING AGENCY NAME AND ADDRESS Washington State Department of Transportation Transportation Building, MS 7370 Olympia, Washington 98504-7370 Project Manager: Gary Ray, 360-705-7975		13. TYPE OF REPORT AND PERIOD COVERED Final research report 14. SPONSORING AGENCY CODE

15. SUPPLEMENTARY NOTES

This study was conducted in cooperation with the U.S. Department of Transportation, Federal Highway Administration.

16. ABSTRACT

The Busview-X project constructed and demonstrated the use of an advanced public transportation system (APTS) to show the viability of providing real-time transit information to transit riders. This project (1) designed an advanced graphical transit information system using data from King County Metro Transit's existing automatic vehicle location (AVL) system and the Puget Sound's regional intelligent transportation systems (ITS) backbone; (2) created a World Wide Web page to launch the application; and (3) demonstrated the system's viability by providing real-time transit coach locations to personal workstations on the University of Washington campus.

As a precursor to an APTS for the Puget Sound region, Busview-X was designed to (1) provide real-time coach location information to the test group; (2) enhance King County Metro's existing investment in AVL technology without disrupting existing operations; (3) evaluate AVL accuracy; (4) encourage increased ridership, modal change, and productivity; and (5) be compatible with federal efforts to develop a national ITS architecture. Busview-X was constructed in an open systems model that supported a distributed computing environment. It used the X-windowing system for graphical support. Busview-X was used 2, 490 times over a period of 670 days from November 1995 to September 1997.

During the Seattle Smart Trek Model Deployment Initiative, the ideas developed in the campus-based version of Busview-X were used to create a new version, Busview, that could be widely supported on the Internet. Busview was written in the Java programming language so that graphical representation of transit information could be available anywhere on the Internet. The incorporation of a graphical toolkit into Java and its inclusion in widely available WWW browsers allowed the researchers to replace the X-window system with WWW browsers as the graphic user interface.

Both versions were well received by users. Comments from Buxview-X were used to improve the design of Busview during the Smart Trek deployment. Refinements to Busview are ongoing.

17. KEY WORDS Automatic vehicle location (AVL), positioning, transit, real-time data, Java applet		18. DISTRIBUTION STATEMENT No restrictions. This document is available to the public through the National Technical Information Service, Springfield, VA 22616	
19. SECURITY CLASSIF. (of this report) None	20. SECURITY CLASSIF. (of this page) None	21. NO. OF PAGES	22. PRICE

Disclaimer

Table of Contents

List of Figures

List of Tables

1. Introduction

This report describes two phases of the development of the Busview project. The first phase was done as a research project and designed principally for University of Washington campus users. The second phase was the construction of a Java applet to make an expanded version of the information available to a regional audience. The report is effectively divided into two major sections. The first section deals with the construction and use of an Advanced Public Transportation System (APTS) as a demonstration of the viability of providing real-time transit information to transit riders on the UW campus. The second section describes the deployment of a Java application that was designed to be accessed from anywhere on the Internet.

Busview-X

The Busview-X project designed and demonstrated a system that displayed real-time transit coach locations on a digital map to the University of Washington campus community. This project (1) designed an advanced graphical transit information system using data from King Count Metro's existing automatic vehicle location (AVL) system and the Puget Sound's regional intelligent transportation systems (ITS) backbone, (2) created a World Wide Web page to launch the application, and (3) demonstrated the system's viability by providing real-time bus location information to individuals at their personal workstations.

As a precursor to an APTS for the Puget Sound region, this system was designed to (1) provide real-time coach location information to the test group; (2) enhance Metro's existing investment in AVL technology without disrupting existing operations; (3) evaluate AVL accuracy; (4) encourage increased ridership, modal change, and productivity; and (5) be compatible with federal efforts to develop a national ITS architecture. BusView-X, was constructed in an open systems model that supported a distributed computing environment. It used the X-windowing system for graphical support.

The Busview application leveraged previous work performed in the Automatic Transit Location System project, sponsored by the Washington State Department of Transportation, to create a widely available application that would provide real-time transit information.

Busview

During the Seattle Smart Trek Model Deployment Initiative (MDI), the ideas developed in the campus-based version of Busview-X were used to create a version that could be widely supported on the Internet. Several technological changes made this wider deployment possible. First, the Java programming language, which facilitates the “write once, run anywhere” philosophy of software authorship, provided a medium that allowed graphical representation of transit information to be made available anywhere on the Internet. Second, World Wide Web (WWW) browsers that can download and execute a Java applet (an interactive program that works inside Web browers on computers and other devices) allowed a wider audience to access Busview. Finally, the incorporation of a graphical toolkit into Java and its inclusion in widely available browsers allowed us to replace the X-window system with WWW browsers as the graphical user interface (GUI). Java in browsers means that powerful GUI support is available on users’ computers at no cost to the user. This set the stage for deploying a working version of Busview that could reach into the individual commuter’s home and work environments.

2. System Architecture

Both versions of the Busview application were built with the component architecture described in Dailey et al [1]. In this model, the GUI is only one component of the application. The component model includes the notion of a data stream flowing sequentially through a series of components that perform data fusion activities on the data stream. The last component in the stream is the GUI . The Busview project focused on the component that the user touches, the GUI (often thought of as the application in other environments), and leveraged the existence of the components from a previous project [2]. This chapter overviews the total application by briefly considering the non-GUI components. It then focuses on the two types of GUI components used in the two stages of the project.

2.1 Application from Components

Busview is a distributed application created within the framework described in Dailey et al [1]. This framework identifies four component types: Source, Redistributor, Operator and Sink, and the Busview application uses all four types.

Busview, shown in Figure 2.1, uses a transit carrier's automatic vehicle location (AVL) system as the sensor for the Source component. In this case, the Source component (labeled *AVLUW*) eavesdrops on communication between components of a proprietary AVL system used by the transit carrier's operations staff to measure schedule adherence and to provide traffic managers with coach locations in the event of an incident. The data in this case are distance along a planned path, which, with knowledge of the routes and several coordinate transformations, can be used to estimate location.

AVL_AVLUW is an Operator component located on a computer in the AVL operations center of the transit carrier. It limits the information that can leave the agency's operations center. In this case, numerical identification of the driver is in the data received by *AVLUW* but is removed for privacy reasons before the data are allowed out of the physical control of the transit agency. This component is also behind the firewalls protecting the transit AVL system from Internet users. Once again, this Operator component provides for network security and agency autonomy, as well as removal of information from the data stream.

The raw data, a distance along a known path, are multiplexed by a Redistributor (labeled *AVL_Rebroadcast*). *AVL_Rebroadcast* multiplexes the original data to make these raw data widely available.

The data from *AVL_Rebroadcast* are passed to an Operator (labeled *AVL_Trip*), which uses information in the AVL data packet to associate a specific vehicle with a specific trip in the transit schedule. The data from the AVL system arrive at an average rate of 11 coaches per second, and a 20,000-record database of scheduled trips is searched for each coach to obtain the corresponding schedule information. This trip information is added to the record for each coach and made available to the downstream clients.

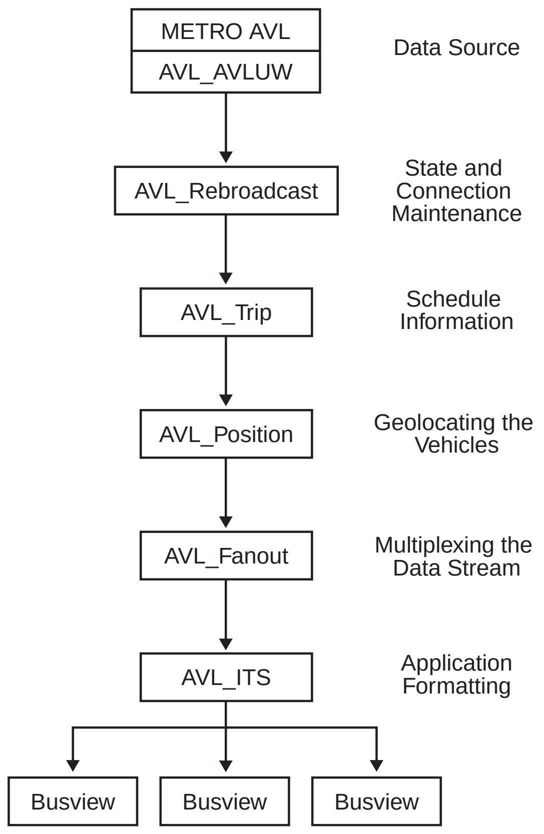


Figure 2.1 Data Flow and Architecture of Busview

The data from *AVL_Trip* are passed to an Operator (labeled *AVL_Position*), which can use the data, information about planned routes, digital maps, and coordinate transformations to calculate a latitude and longitude value for the transit (probe) vehicles. The calculation of position requires (1) identifying the particular coordinate lists in one of 2,500+ files that represent every possible vehicle route, (2) selecting the segment within the list on which the vehicle is estimated to exist, and (3) using a Newton's method to perform an inverse Lambert projection from the proprietary coordinate system of the AVL system, at a rate of 11 vehicles per second. We designed into our approach the possibility of components operating on independent CPUs so that we can select a CPU of appropriate power for each of the individual tasks. This Operator, for example, takes the entire resources of one computer to keep up with the data flow, and therefor it does not coexist on a CPU with the other Operators in these examples.

Downstream of the *AVL_Position* server is a Redistributor labeled *AVL_Fanout*, which multiplexes the data stream containing the vehicle positions to a variety of users. In addition to performing the positioning solution, this Redistributor frees the upstream process from multiplexing.

The server component labeled *AVL_Its* creates a very customized data stream for the Busview presentation. The connection between this server and the Busview sink can potentially be over slower media. This component minimizes the amount of data that need be sent per update of each transit vehicle on the Busview screen. A digital map and the vehicle location information, along with schedule information, are displayed graphically on the Busview screen, creating an information system for transit riders. The Busview graphical user interface shown in Figure 2.2 operates in an X-Window environment. It is a distributed application example created within the ITS framework [1]. The Busview GUI produces a representation of map data on an X-terminal that may be located anywhere on the Internet.

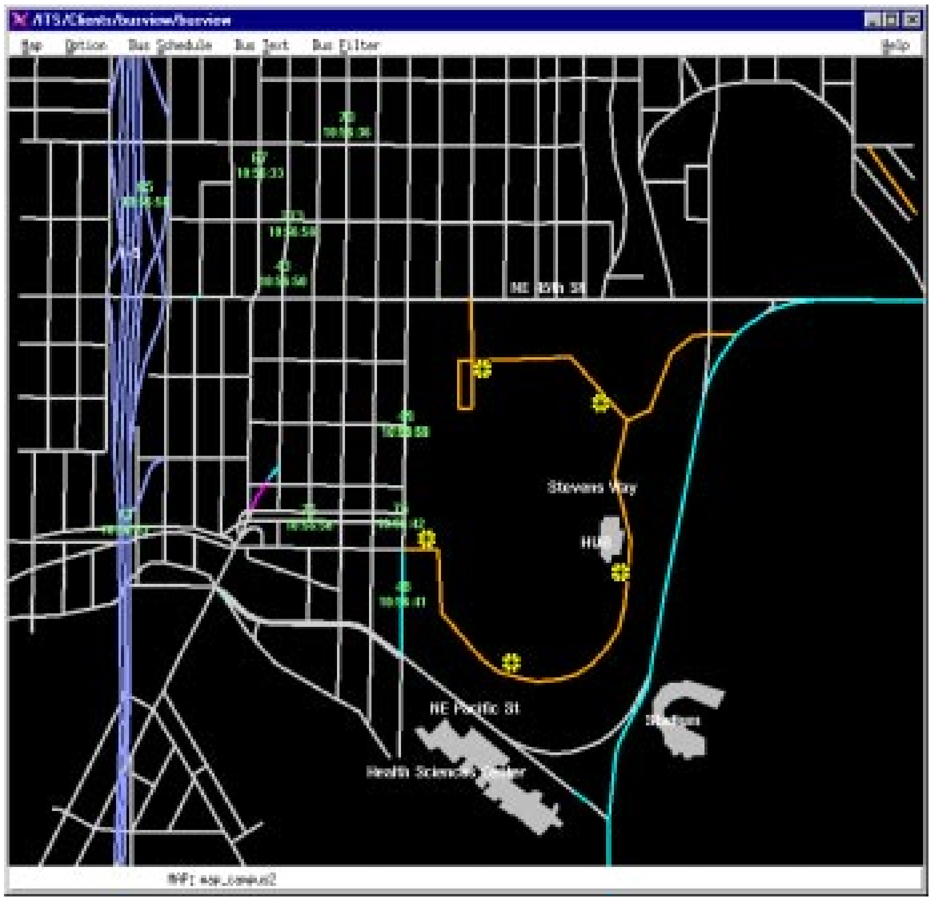


Figure 2.2 Busview graphical user interface

2.2 Graphical User Interface: Busview-X

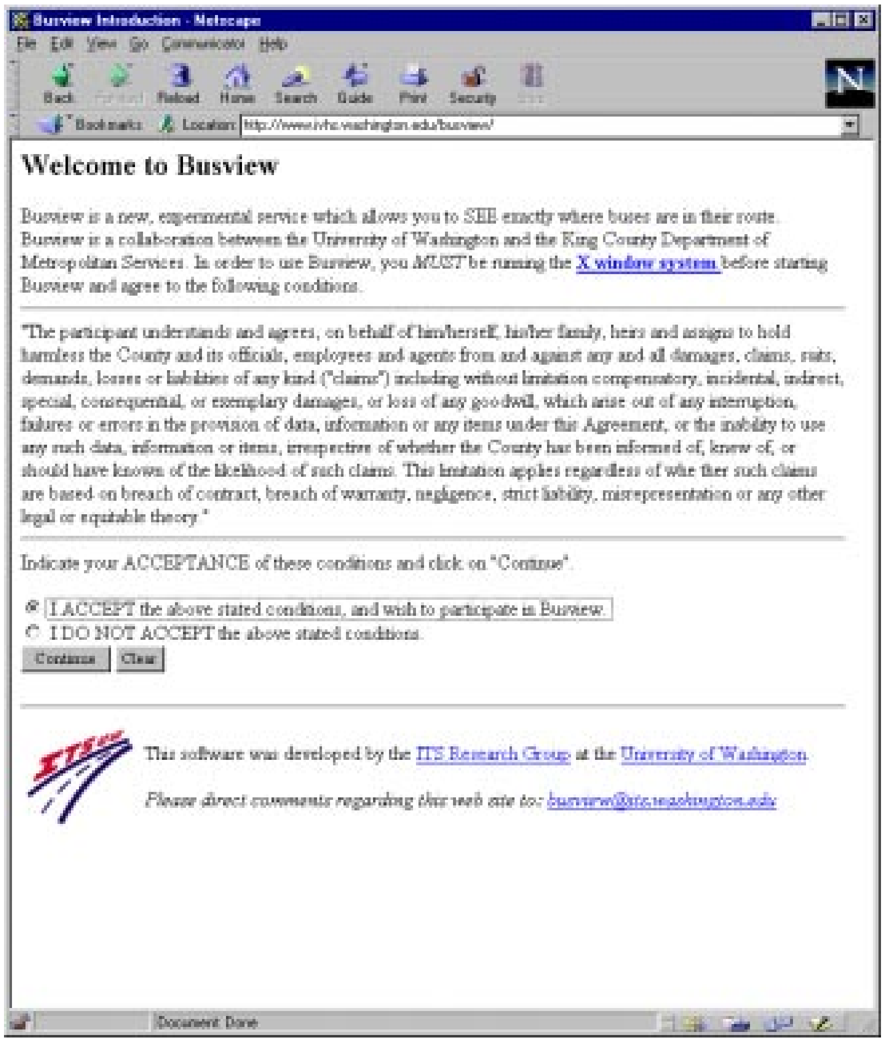
The Busview project created a GUI that would allow anyone on the Internet who operated an X-windows display server to display transit vehicle locations in real time. Making real-time transit information publicly available had customer service impacts for the transit agency. Therefore, project participants decided that the initial version would be experimental and deployed only on the UW campus. In addition, the transit agency's council required that any users accept the following conditions:

> *"The participant understands and agrees, on behalf of him/herself, his/ her family, heirs and assigns to hold harmless the County and its officials, employees and agents from and against any and all damages, claims, suits, demands, losses or liabilities of any kind ("claims") including without limitation compensatory, incidental, indirect, special,*

consequential, or exemplary damages, or loss of any goodwill, which arise out of any interruption, failures or errors in the provision of data, information or any items under this Agreement, or the inability to use any such data, information or items, irrespective of whether the County has been informed of, knew of, or should have known of the likelihood of such claims. This limitation applies regardless of whether such claims are based on breach of contract, breach of warranty, negligence, strict liability, misrepresentation or any other legal or equitable theory."

Application Launch/Web Browser

To facilitate the use of the Busview application, a World Wide Web (WWW) page was created to launch the X-windows application (see Figure 2.3). The first WWW page required the user to acknowledge the conditions of use required by the transit agency. A subsequent


Busview Introduction - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Location: http://www.its.washington.edu/busview/
Document: Done

Welcome to Busview

Busview is a new, experimental service which allows you to SEE exactly where buses are in their route. Busview is a collaboration between the University of Washington and the King County Department of Metropolitan Services. In order to use Busview, you *MUST* be running the X window system before starting Busview and agree to the following conditions.

"The participant understands and agrees, on behalf of him/herself, his/her family, heirs and assigns to hold harmless the County and its officials, employees and agents from and against any and all damages, claims, suits, demands, losses or liabilities of any kind ("claims") including without limitation compensatory, incidental, indirect, special, consequential, or exemplary damages, or loss of any goodwill, which arise out of any interruption, failures or errors in the provision of data, information or any items under this Agreement, or the inability to use any such data, information or items, irrespective of whether the County has been informed of, knew of, or should have known of the likelihood of such claims. This limitation applies regardless of whether such claims are based on breach of contract, breach of warranty, negligence, strict liability, misrepresentation or any other legal or equitable theory."

Indicate your ACCEPTANCE of these conditions and click on "Continue".

[✓ I ACCEPT the above stated conditions, and wish to participate in Busview.]
I DO NOT ACCEPT the above stated conditions.

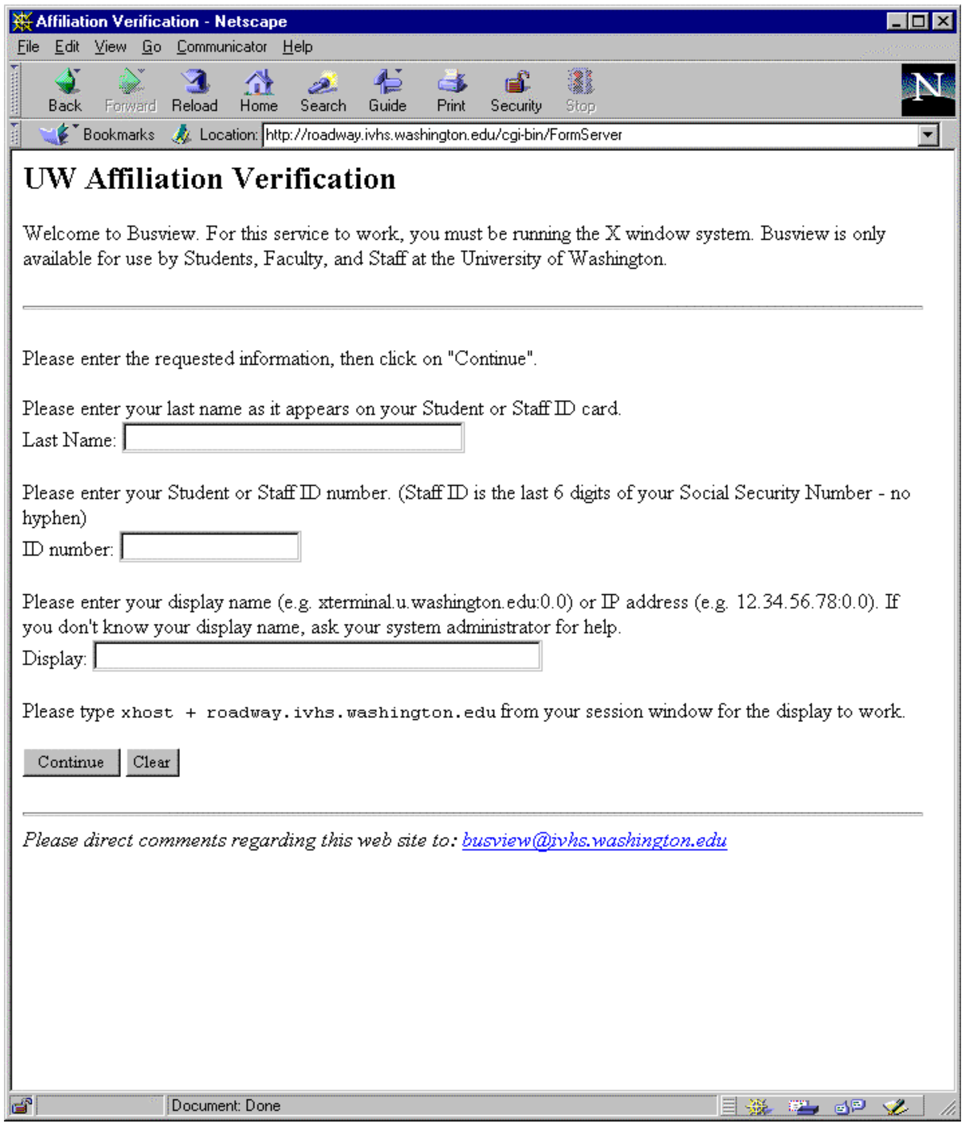
Continue Clear

This software was developed by the ITS Research Group at the University of Washington.

Please direct comments regarding this web site to: busview@its.washington.edu

Figure 2.3 Busview Homepage

page (see Figure 2.4) required the user to enter a name and UW identification number that was checked against central UW records to verify that the user was a member of the UW faculty/ student/staff community. This verification was done with a common gateway interface (CGI) “C” language program written for this purpose. If authentication succeeded, a map graphic was displayed on the user’s workstation (see Figure 2.2) with the X-window system, and subsequent transit vehicle information was plotted on this map.


Affiliation Verification - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Location: http://roadway.ivhs.washington.edu/cgi-bin/FormServer
Document: Done

UW Affiliation Verification

Welcome to Busview. For this service to work, you must be running the X window system. Busview is only available for use by Students, Faculty, and Staff at the University of Washington.

Please enter the requested information, then click on "Continue".

Please enter your last name as it appears on your Student or Staff ID card.
Last Name:

Please enter your Student or Staff ID number. (Staff ID is the last 6 digits of your Social Security Number - no hyphen)
ID number:

Please enter your display name (e.g. xterminal.u.washington.edu:0.0) or IP address (e.g. 12.34.56.78:0.0). If you don't know your display name, ask your system administrator for help.
Display:

Please type `xhost + roadway.ivhs.washington.edu` from your session window for the display to work.

Continue Clear

Please direct comments regarding this web site to: busview@ivhs.washington.edu

Figure 2.4 UW Affiliation Verification Page

Map Support

The map database for this project was derived from a digital map created by the King County Metro GIS division. The geo-location data about roadways were converted from the METROKC format (ASCII) to a data structure developed in this project and stored as a much smaller binary file. The Busview display application used this locally developed binary file structure to create the map graphic. The data format was converred for two reasons: (1) the structure of the binary information improved the speed of accessing items that were part of the map, and (2) the structure reduced the memory requirements placed on the display client when the system displayed either maps of a large area or maps containing a high density of information (e.g., an urban area with all the streets, arterials, and highways).

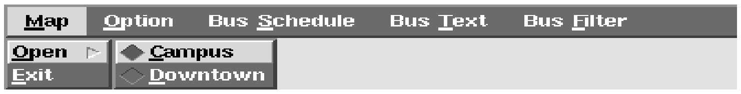
User Interface

The Busview client read a binary map file and displayed the map on an X-terminal whose address was determined at the bottom of the Web page shown in Figure 2.2. Real-time transit vehicle locations were obtained by establishing a connection to the AVL_IVHS information server component shown in Figure 2.1. The transit vehicles were represented by an icon containing the service route number and time stamp indicating when the transit vehicle was observed at the location shown. When a vehicle was observed more than once, an arrow was added to the icon to indicate direction.

In addition to the transit service route number, the user interface provided some simple GIS functions, such as

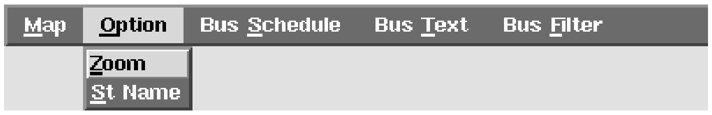
- selection of different areas (UW campus and downtown Seattle)
- zoom in
- point to a feature and get information (street names and bus schedules)
- adjust the bus text font shown in the map
- select the bus route numbers.

The menu bar displayed all the available functions supported by the Busview system. Each of these menu bar items had a pull-down menu. For example, the Map menu had the following:



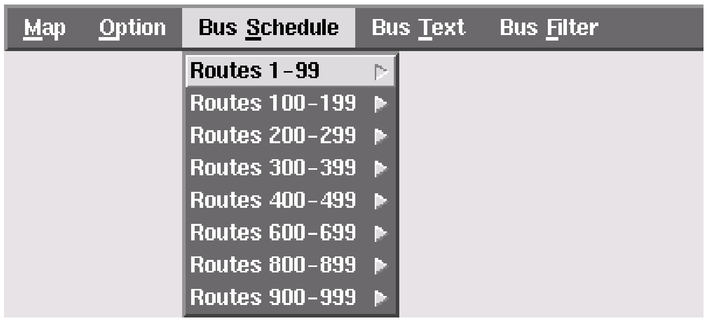
- Open - This selection opened a new display of the map files. Users could choose a map of either the UW campus or the downtown area.
- Exit - This selection exited the Busview program.

The Option menu had the following:



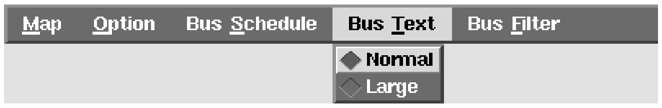
- Zoom - This option allowed users to zoom in on a specific area of the map by creating a rectangular area. Users could return to a larger viewing area by refreshing the screen with the Open selection under the Map pull-down menu.
- St Name - This option produced a pop-up screen that allowed users to place street names on the map.

The Bus Schedule menu allowed access to the static schedule information through a Routes pull down:



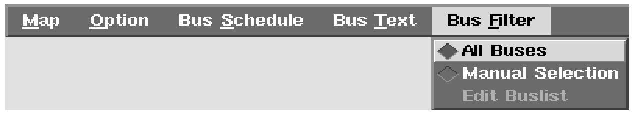
- Routes - Users could display a printed time schedule for a specific bus route. Following the arrows to the right of each category narrowed the selection, and the schedule appeared in another window on the screen. Users could also produce a time schedule for a bus route by clicking directly on the bus number where it appeared on the map.

The Bus Text menu item selected the font sizes for the text in the icon representing the transit vehicle:



- Normal - This was the default option for displaying the bus numbers on the map. The numbers were shown in 12 pixels format.
- Large - This option allowed the bus numbers to be displayed in a larger type format of 18 pixels.

Finally, the Bus Filter menu item allowed users to select which transit service route would appear on the map display:



- All Buses - This was the default selection and displayed all buses that were transmitting location information.
- Manual Selection - This option narrowed the display to specific bus routes. A pop-up screen allowed users to enter a route number and then click on the Add button to create a custom list of routes to view. Clicking on the OK button displayed them on the map.
- Edit Buslist - This option either added or removed bus routes after a list had been created with the Manual Selection option.

Source Code File Description

This section describes each of the source files (written in the C and Tcl/Tk languages) that, along with the Makefile that controlled the compilation and linking of the applications, made up the software developed for the Busview-X client system.

Tcl/Tk was used to build the Busview-X client system because it is a powerful programming environment for creating graphical user interfaces. Tcl stands for Tool Command Language, and Tk is its associated X-windows toolkit. Tcl/Tk provides a rapid development environment and many sophisticated built-in features. The communication and map retrieval portions of Busview were implemented in the C language, and the GUI portion was implemented in the Tcl/Tk language.

C files

main.c:	This ws the central routine for the Busview client system. This routine created a Tcl interpreter and defined Tcl commands for the Busview application.
rdbf.c:	This routine read binary map data from a file to memory.
draw.c:	This file contained subroutines to process events, such as resize and zoom-in, as well as to draw or redraw when needed.

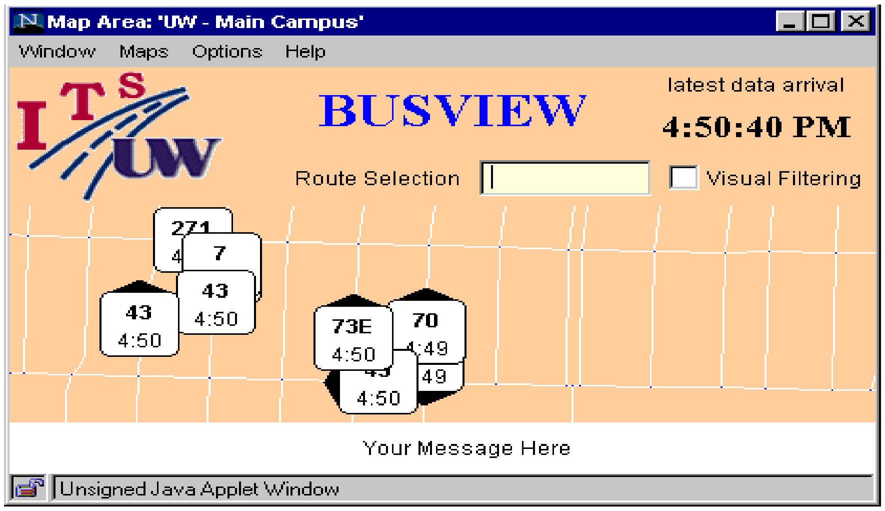
socket.c:	This routine handled the communication with the AVL_IVHS server system to get the real-time Metro transit vehicle locations.
bus.c:	This routine processed the data received from the AVL_IVHS server. It decided whether the bus was in or out of the map area. It also transformed the latitude and longitude of the bus location into the coordinate system used for the display.
street.c:	This routine searched the map database and returned the name of the street that had been selected by the user.

Tcl files

map.tcl:	This file containd the initial setup of the Busview-X system display, such as window height, window width, foreground and background color, menubar location, and contents of the menubar.
bus.tcl:	This file contained the routines to display the bus route number, time stamp, and bus direction on the map. It also contained the routine to remove vehicles that no longer appear in the data stream.
bussize.tcl:	This routine resized the bus route number font.
businfo.tcl:	This routine showed the bus schedule selected by the user.
buslist.tcl:	This routine created the list of routes the user wished to see on the map.
help.tcl:	This routine contained the help contents for the Busview application.
street.tcl:	This routine showed the street name selected by the user.
landmark.tcl:	This routine displayed the landmarks (buildings, hospital, and stadium) on the map.
zoom.tcl:	This file included subroutines to select a sub-map from an existing map window and create a new map based on the subarea selected. It used rubber lines when selecting a sub-area from the map window.

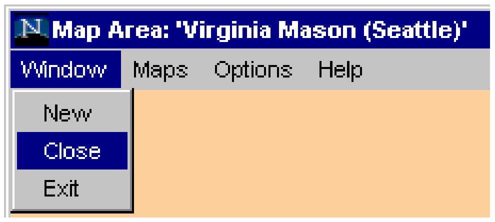
2.3 Graphical User Interface: Busview

Introduction



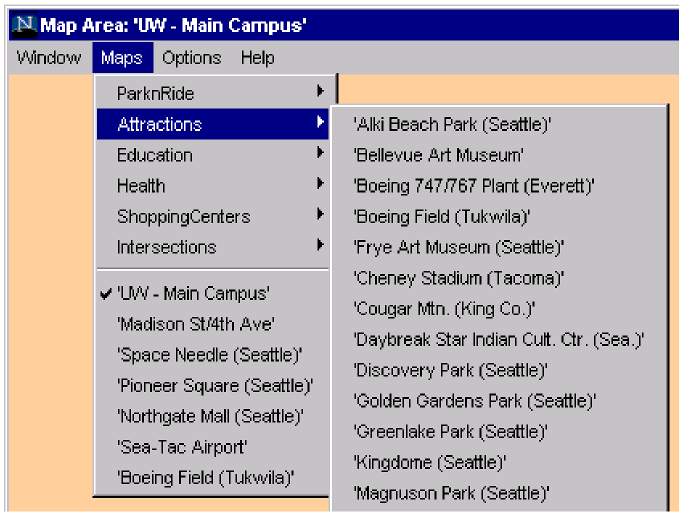
The image above is a generic screen shot of the Busview applet showing the default map of the University of Washington campus. Users can select another location from maps that cover the King County area. Each black and white icon on the map represents a different bus moving along its route. In the upper right corner is a time display that shows the last time the Busview applet received data. The route selection box at the top allows users to limit the number of bus routes being displayed on the map.

Window Menu



There are three options under the Window menu: new, close, and exit. The “new” option allows a user to have multiple windows open for various locations. By choosing “new” and then selecting a different location from the Maps menu, a user can create as many maps as needed. The "close" option closes the current window. The "exit" option closes all open windows, thus shutting Busview down completely.

Maps Menu

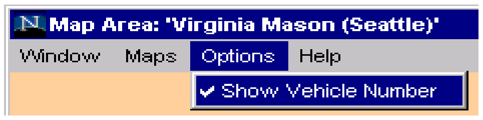

Map Area: 'UW - Main Campus'
Window
Maps
Options
Help
ParknRide
Attractions
Education
Health
ShoppingCenters
Intersections
✓ 'UW - Main Campus'
'Madison St/4th Ave'
'Space Needle (Seattle)'
'Pioneer Square (Seattle)'
'Northgate Mall (Seattle)'
'Sea-Tac Airport'
'Boeing Field (Tukwila)'
'Alki Beach Park (Seattle)'
'Bellevue Art Museum'
'Boeing 747/767 Plant (Everett)'
'Boeing Field (Tukwila)'
'Frye Art Museum (Seattle)'
'Cheney Stadium (Tacoma)'
'Cougar Mtn. (King Co.)'
'Daybreak Star Indian Cult. Ctr. (Sea.)'
'Discovery Park (Seattle)'
'Golden Gardens Park (Seattle)'
'Greenlake Park (Seattle)'
'Kingdome (Seattle)'
'Magnuson Park (Seattle)'

The Maps menu allows the user to cascade through a series of pull-down menus to select a region of King County one mile wide. Once the selection has been made, a new map is drawn. Buses appear as they move onto the map being viewed. A user can have multiple windows open for various locations by choosing "new" from the Window menu and then selecting a different location from the Maps menu.

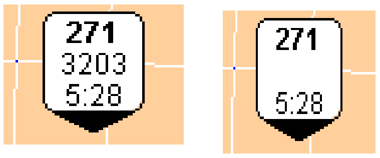
The default map selection is for the University of Washington campus in Seattle. Several maps have been pre-loaded for easy selection. A history list remembers the last seven maps that were viewed. When a new map is displayed, it moves to the top of the history list. The current map has a check mark beside it, and when the history list is greater than seven, the oldest map drops off the list.

In testing, we have found the performance of the cascading menu to be very poor with Netscape Navigator on the Windows platform. This is a result of the Navigator browser and not the applet. Running the applet in Internet Explorer does not produce the same slowdown effect.

Options Menu

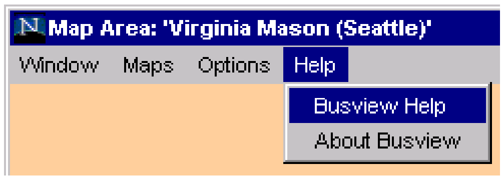


Currently, only one option is available under this menu. This option toggles the vehicle identification number (VIN) on and off for the bus location icon. The VIN is a number that is painted on the back and top of each bus. As other options are available, they will be listed under this pull-down menu.



The image on the left shows the bus location icon with the VIN on, and the second image shows it with the VIN off. The default for the main maps is for the VIN to be off. When the bus progress bar is being viewed, the VIN is always on. This allows the user to track the correct bus along an entire route.

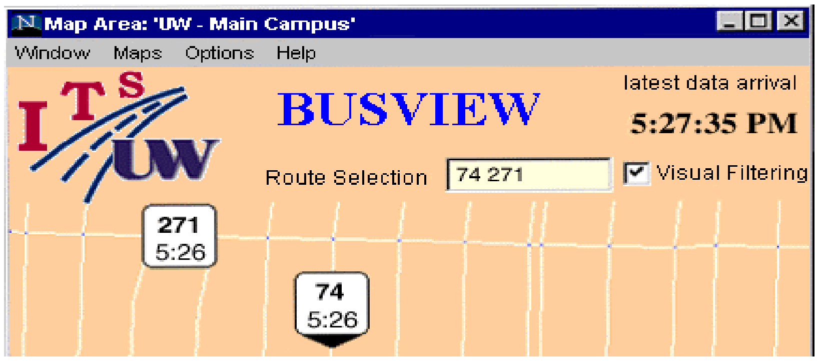
Help Menu



Choosing “Busview Help” links the user’s browser to this Web page. Choosing “About Busview” displays the Busview version number.

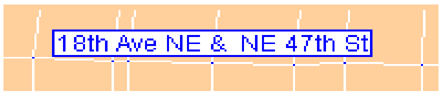
Route Selection/Visual Filtering

Users can select which bus routes to view by checking the “Visual Filtering” box and entering one or more bus routes separated either by commas or spaces. This will filter out other bus routes, allowing users to monitor only routes of immediate interest. In the image below, two routes have been entered in the “Route Selection” box. This may produce a blank screen



if no buses are running on the selected route(s). Once buses from the selected route(s) enter the map area, however, they will be visible.

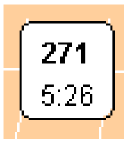
Street Names

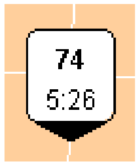


Intersections on the map have blue "hot spots." When the cursor passes over them, a street name pops up, as seen in the above image, .

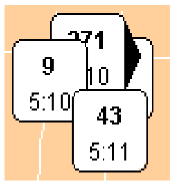
Bus Location Icon

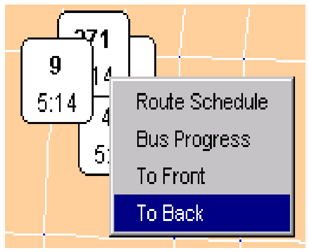
These icons represent the buses and appear on all map screens. The top number (in bold type) is the route number of the bus. The bottom number is a time stamp showing the last time that the bus was heard from. This time stamp should be compared with the "latest data arrival" time display in the upper right corner of the applet window to check the timeliness of the bus data.





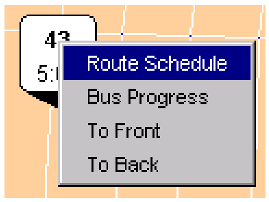
The black "arrow" on the second image shows the direction the bus is traveling along its route. If no arrow is showing, the bus has not moved since it appeared on the screen. Once the bus starts moving along its route, an arrow appears to indicate its direction.





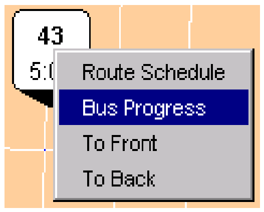
At times, some bus icons may be partially hidden because of the number of buses showing on the screen. Clicking on a bus location icon will bring up four additional options. The last two options, "To Front" and "To Back," allow the user to choose which icon to view. In the images above, clicking on "To Front" will bring the selected bus icon to the top of the stack, and choosing "To Back" will send it to the bottom of the stack.

Route Schedule

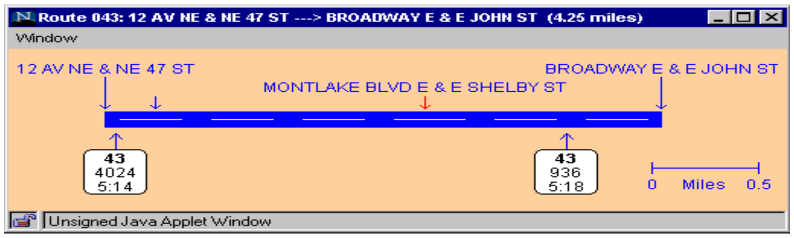


After clicking on a bus location icon, one option is "Route Schedule," as shown above. Selecting this option links the user's browser to the Metro bus schedule for that bus route.

Bus Progress

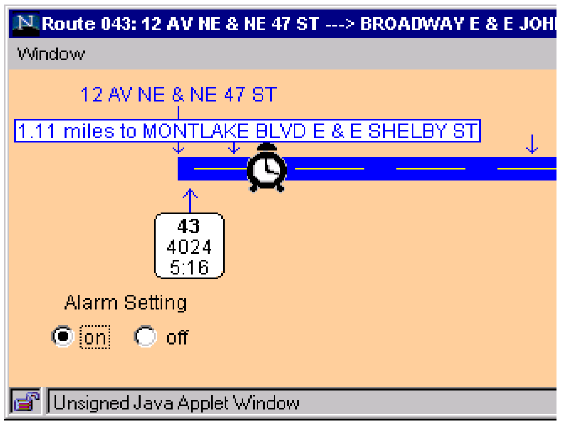


After a bus location icon has been selected, the "Bus Progress" option allows the user to see a visual, linear representation of the chosen bus route with the coach location(s) along it. The blue arrows represent intersections and work the same as the intersection hot spots on the main display. When the cursor passes over an intersection, the intersection's name pops up for viewing, and the corresponding arrow turns red.


Route 043: 12 AV NE & NE 47 ST ---> BROADWAY E & E JOHN ST (4.25 miles)
Window
12 AV NE & NE 47 ST
BROADWAY E & E JOHN ST
MONTLAKE BLVD E & E SHELBY ST
43
4024
5:14
43
936
5:18
0
Miles
0.5
Unsigned Java Applet Window

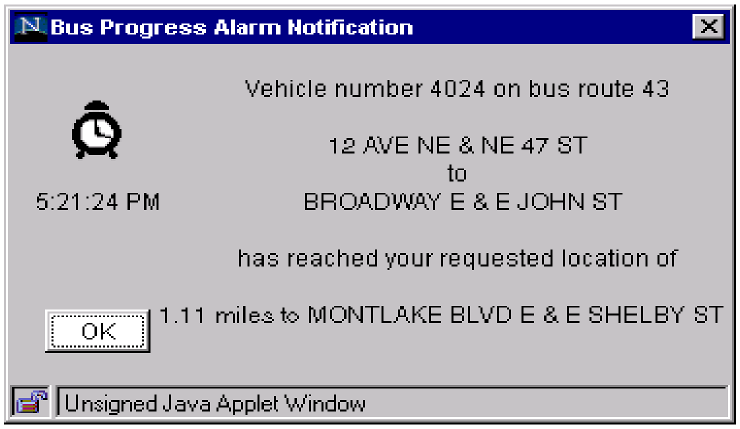
Buses move from left to right on the screen. The beginning and ending points for the route are listed both at the ends of the blue "roadway" and at the top of the window bar.

Alarm Setting


Route 043: 12 AV NE & NE 47 ST ---> BROADWAY E & E JOH
Window
12 AV NE & NE 47 ST
1.11 miles to MONTLAKE BLVD E & E SHELBY ST
43
4024
5:16
Alarm Setting
on
off
Unsigned Java Applet Window

An alarm can be set along any progress bar by clicking on the blue "roadway." The alarm notifies users when a bus has passed a particular point on the route. This allows users to better estimate their departure time when catching a bus. Since the buses move from left to right, the alarm must be set to the right of the chosen bus to be effective. Only one alarm may be set on each progress bar. After the roadway is clicked , an alarm clock will show on the screen, along with information about where it has been set, as seen in the image above.

Once the bus has reached the alarm setting, a notification window appears (see below). The alarm also emits an audible reminder when the bus passes. The "Alarm Setting" in the progress window is then set to "off." The alarm will not go off again if another bus on the progress bar reaches the alarm location. Manually setting the alarm status to "on" will re-prime the alarm and produce further notifications.



Bus Progress Alarm Notification

Vehicle number 4024 on bus route 43

12 AVE NE & NE 47 ST
to
BROADWAY E & E JOHN ST

5:21:24 PM

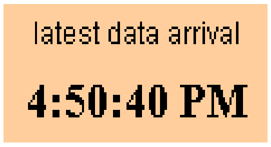
has reached your requested location of

1.11 miles to MONTLAKE BLVD E & E SHELBY ST

OK

Unsigned Java Applet Window

Latest Data Arrival Time Display



latest data arrival

4:50:40 PM

The time display in the upper right of the window shows when data were last received by the Busview applet. Information comes approximately every second. Data packets do not contain information on every bus, as new data are available for each bus approximately every minute. Comparing the time display to the time on a bus icon helps indicate the timeliness of the information. If the disparity between the two times is large (i.e., the bus time is 2:55 and the display time is 3:00), then the bus information is out of date. The icon shows the last KNOWN position of a bus, and until the bus sends more data, the image is stationary. As a result, the buses appear to “hop” across the screen. Busview presently shows only current data and makes no attempt to predict bus locations.

3. Usage Statistics and Survey Responses

3.1 Usage Statistics for Busview-X

The Busview-X application ws launched by a WWW page whose URL was <http://www.its.washington.edu/busview>. The httpd server that launched Busview-X maintained statistics on Busview-X usage, and those statistics are reported here. The Busview-X application was used 2,490 times over a period of 670 days from November 1995 to September 1997. Figure 3.1 shows Busview-X's use as a function of time. The high intial use corresponds to a period when the University Homepage highlighted the project (December-February 1996). There were 439 distinct computers that used Busview. Records of individual use patterns were not kept.

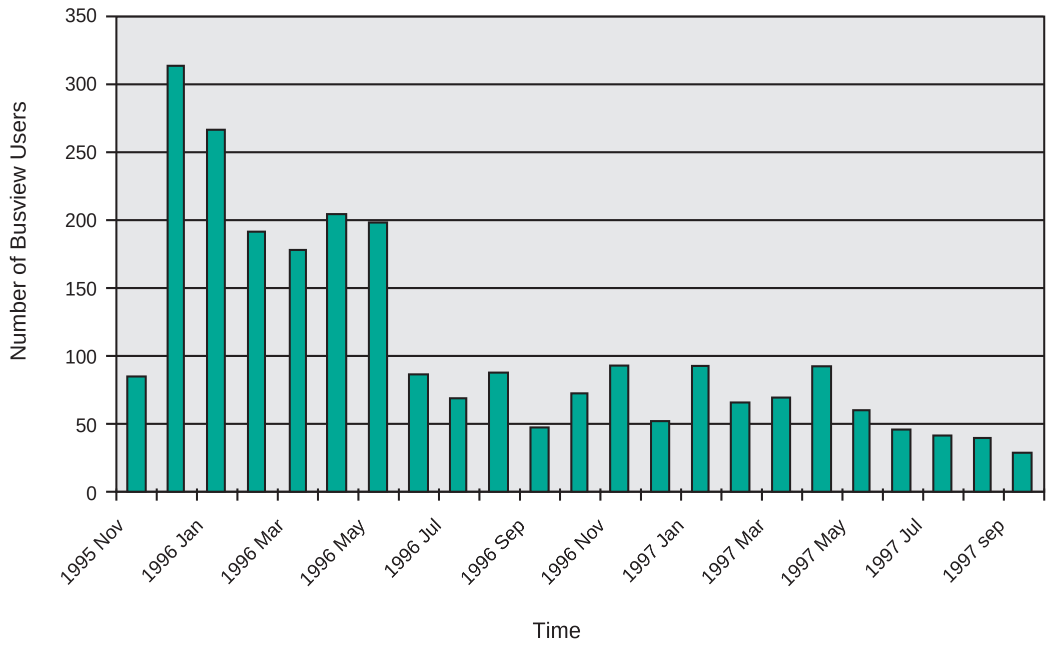
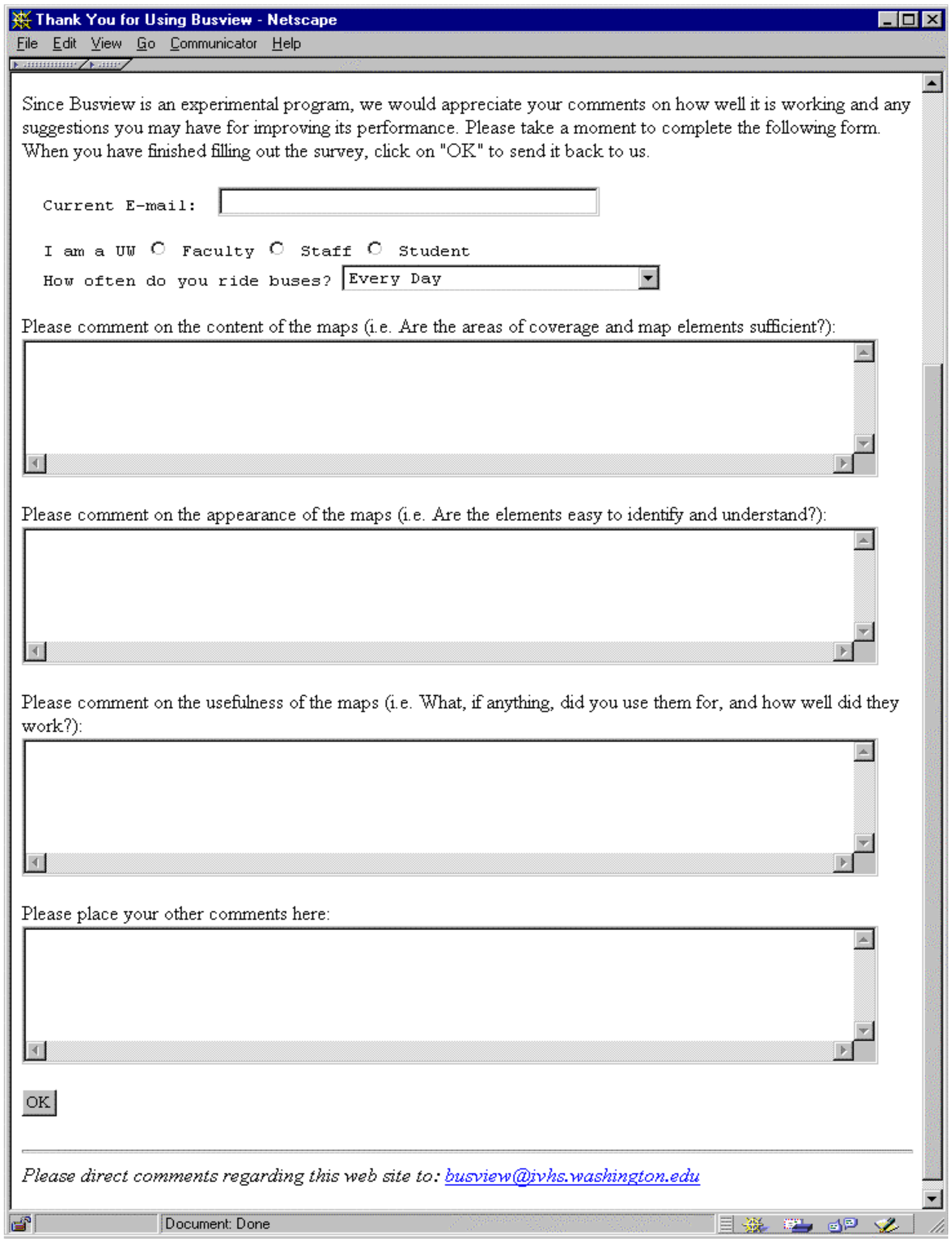


Figure 3.1 Number of Busview uses as a function of time


Thank You for Using Busview - Netscape
File Edit View Go Communicator Help

Since Busview is an experimental program, we would appreciate your comments on how well it is working and any suggestions you may have for improving its performance. Please take a moment to complete the following form. When you have finished filling out the survey, click on "OK" to send it back to us.

Current E-mail:

I am a UW ○ Faculty ○ Staff ○ Student

How often do you ride buses? Every Day

Please comment on the content of the maps (i.e. Are the areas of coverage and map elements sufficient?):

Please comment on the appearance of the maps (i.e. Are the elements easy to identify and understand?):

Please comment on the usefulness of the maps (i.e. What, if anything, did you use them for, and how well did they work?):

Please place your other comments here:

OK

Please direct comments regarding this web site to: busview@ivhs.washington.edu

Figure 3.2 Busview survey page

3.2 Survey Results for Busview-X

A survey was displayed after the Busview-X application was launched. Respondents were free to respond to any or all of the questions in the survey instrument, shown in Figure 3.2. The first question requested an email address; question two requested their UW status (faculty/student/staff); and question three inquired about their transit use pattern. A total of 115 responded to the request for UW status. Of these, the majority were students, half as many were staff, and a small percentage were faculty, as shown in Figure 3.3. A total of 217 users responded to question three concerning the frequency of transit use, and Figure 3.4 indicates that the majority of the people interested enough to use the application were regular transit users. It is noteworthy that the users were not forced to fill in a survey, and so, of the 2,490 uses by 439 individuals, we received a 26 percent response-per-user rate (115) to the status reports and a 49 percent response-per-user rate (217) to the transit frequency use reports. Clearly, many of the uses did not result in survey information.

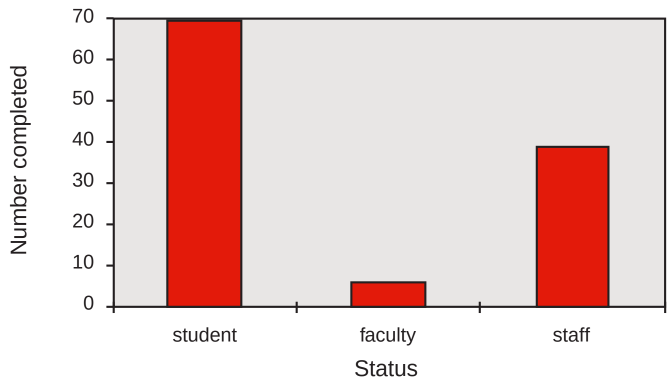


Figure 3.3 Users' UW status

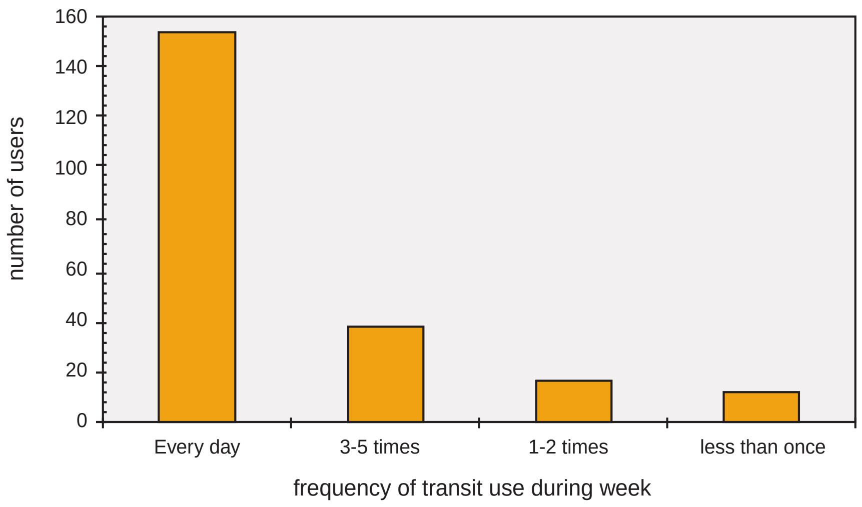


Figure 3.4 Frequency of transit use

In addition to the specific items above, we asked the users three open-ended questions:

1) Are the areas of coverage and map elements sufficient?
2) Are the elements easy to identify and understand?
3) What, if anything, did you use them for, and how well did they work?
4) Other comments.

The comments from the users are included in Appendix A.

3.3 Usage Statistics for Busview

In using the Busview Applet developed in SmartTrek, users first download the Java applet (referred to as a JAR file) and then execute it. The applet creates a connection to the Busview server through which the data stream containing bus location information is obtained. As a result, the use statistics for the Busview Java applet have two aspects: (1) the number of times the applet is downloaded, and (2) the number of times the data stream is opened, indicating an active session. These two statistics differ because Internet browsers cache the applet. (e.g., once the applet is downloaded, it is stored on the disk where the browser resides for some period of time, and any time the applet is referenced, the browser uses the local copy rather than downloading a new copy). The two subsections below present the statistics for these two activity measures from intial deployment of the Busview applet to the date of this writing.

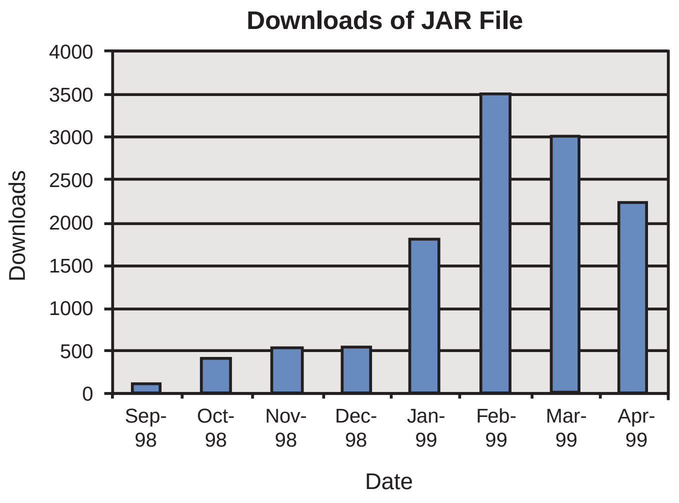


Figure 3.5 JAR downloads by months

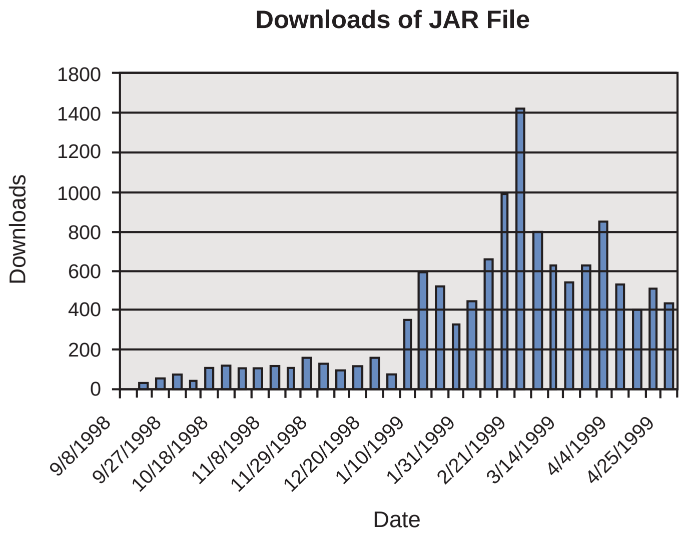


Figure 3.6 Downloads by weeks

3.3.1 Use Statistics for Busview Applet Download

Statistics for downloading the Busview applet (a JAR file) were collected from September 1998 to April 1999. Over that period, there were 12,300 successful downloads, averaging 53 per day. These requests were made from 6,115 distinct hosts. The download rate changed with time as the applet was discovered and used. In Figure 3.5, the downloads as a function of month are shown. Figure 3.6 shows the number of downloads each week over the

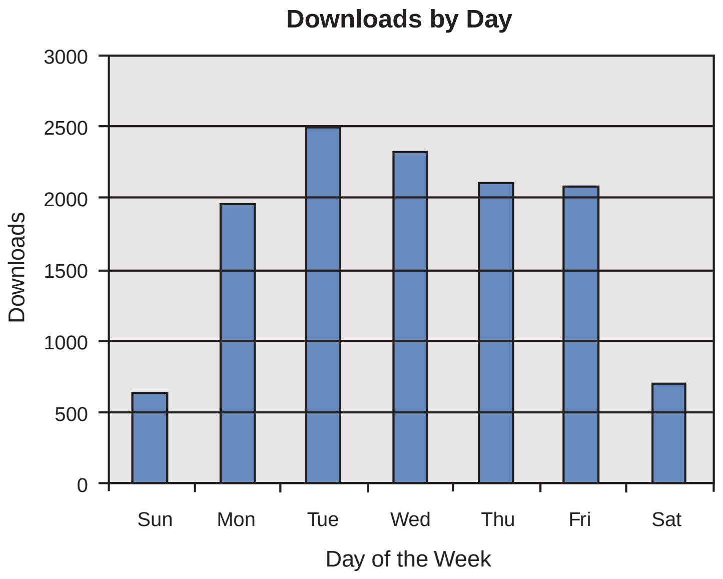


Figure 3.7 Downloads by day

reporting period. It is noteworthy that two events seem to be reflected in the use pattern: (1) a link to Busview appeared on the site <http://www.scripting.com>, a popular site among programmers, on February 23, 1999, and (2) signs appeared on Metro buses around March 31, 1999. Both of these events seem to have increased use immediately afterward. Figure 3.7 shows the use by day of week, indicating that the applet is more heavily used during the work week than on weekends. Figure 3.8 shows the downloads by time of day and suggests that the application is more frequently used during the evening commute. This last observation, about afternoon commuter use, is augmented by the observation that the use, broken down by Internet domain (shown in Table 3.1), is heavily weighted toward commercial users (the ".com" domain). This suggests that users may be using Busview to plan trips from work to home during the afternoon rush hours.

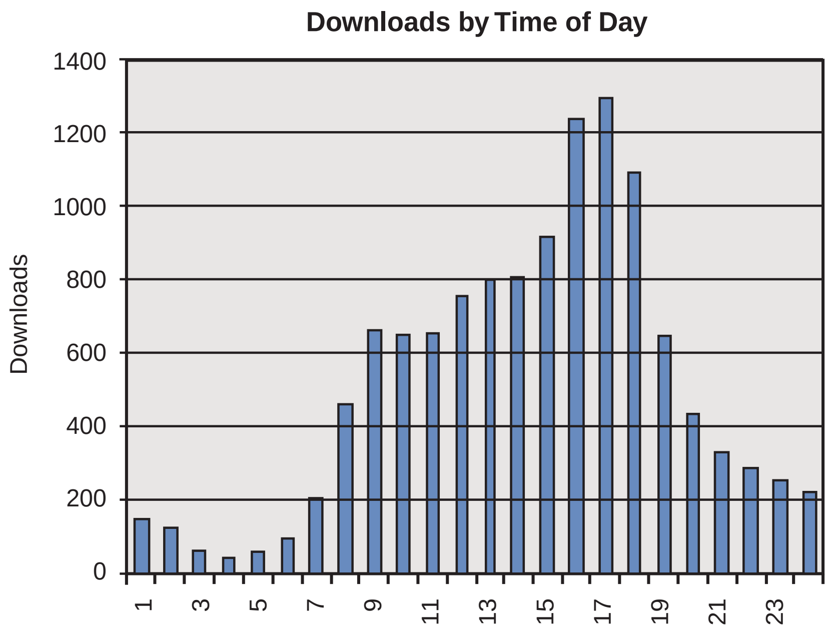


Figure 3.8: Downloads by time of day

Table 3.1 Internet Domains downloading the Busview applet, sorted by the amount of traffic

Number	Domain
4264	.com (Commercial)
2591	.edu (USA Educational)
2397	[unresolved numerical addresses]
2260	.net (Network)
273	.nl (Netherlands)
136	.us (United States)
125	.org (Non-Profit Making Organizations)
86	.gov (USA Government)
30	.ca (Canada)
21	.jp (Japan)
19	.se (Sweden)
19	.uk (United Kingdom)
10	.de (Germany)
9	.mil (USA Military)
9	.au (Australia)
7	.be (Belgium)
6	.sg (Singapore)
7	.hk (Hong Kong)
4	.es (Spain)
4	.dk (Denmark)
5	.fi (Finland)
5	.it (Italy)
3	.nz (New Zealand)
2	.ie (Ireland)
1	.qa (Qatar)
1	.kr (South Korea)
1	.ar (Argentina)
1	.is (Iceland)
1	.at (Austria)
1	.tw (Taiwan)
1	.mx (Mexico)
1	.th (Thailand)

3.3.2 Use Statistics for Busview Data Stream

Statistics for the use of the Busview data stream are available from April 1, 1999. This section summarizes the use of the data stream from April 1, 1999 to May 1, 1999. During the month of April, there were 3,826 successful uses of the data stream, averaging 153 uses per day by 1,386 distinct computers. Figure 3.9 shows the number of uses each week over the reporting period. Figure 3.10 shows the number uses each day for the reporting period. Figure 3.11 shows the use by day of week, indicating that the applet is more heavily used during the work week than on weekends. Figure 3.12 shows the data stream usage by time of day, and

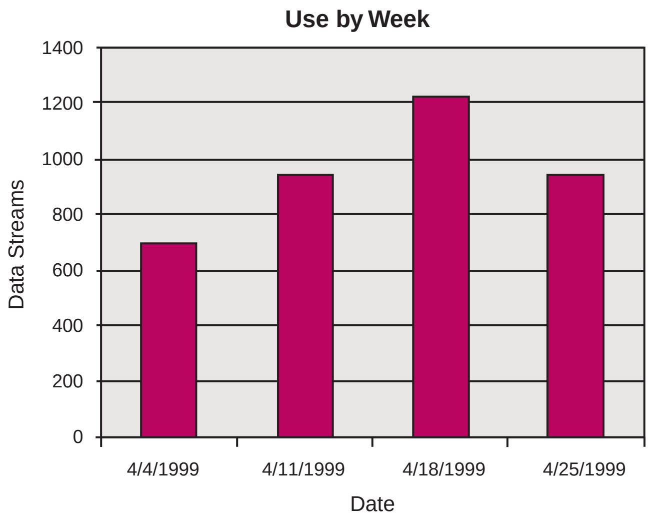


Figure 3.9: Data stream connections by week

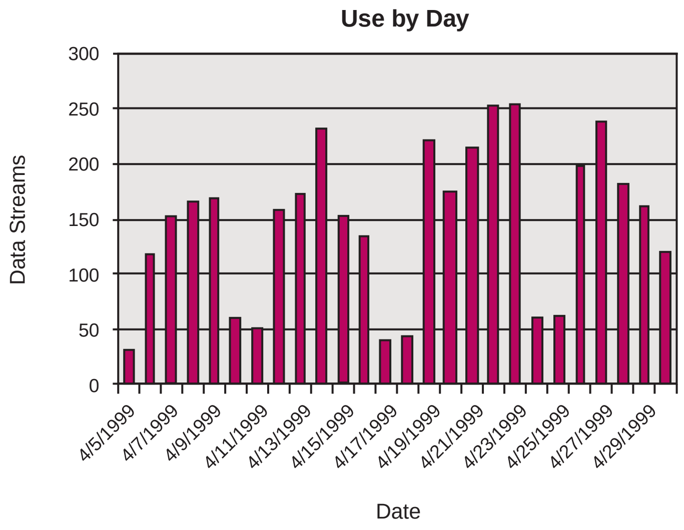


Figure 3.10: Data Stream connections by day

again suggests that the application is more frequently used during the evening commute. This last observation, about afternoon commuter use, is augmented by the observation that the use, broken down by Internet domain (shown in Table 3.2), is heavily weighted toward commercial users (the ".com" domain). This suggests that users may be accessing Busview to plan trips from work to home during the afternoon rush hours.

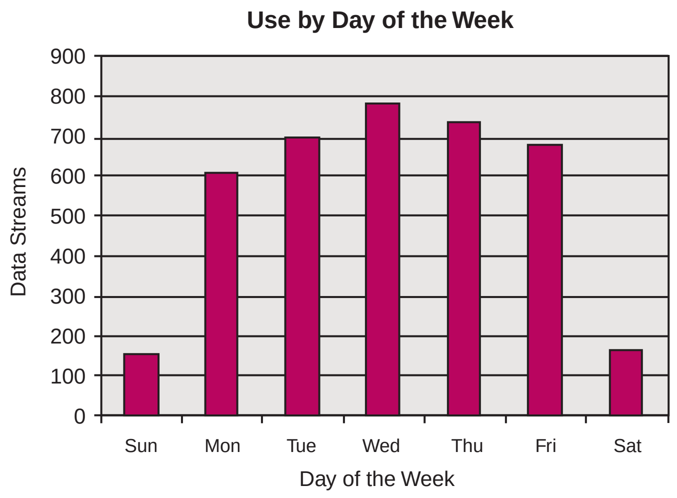


Figure 3.11: Data stream connection by day of the week

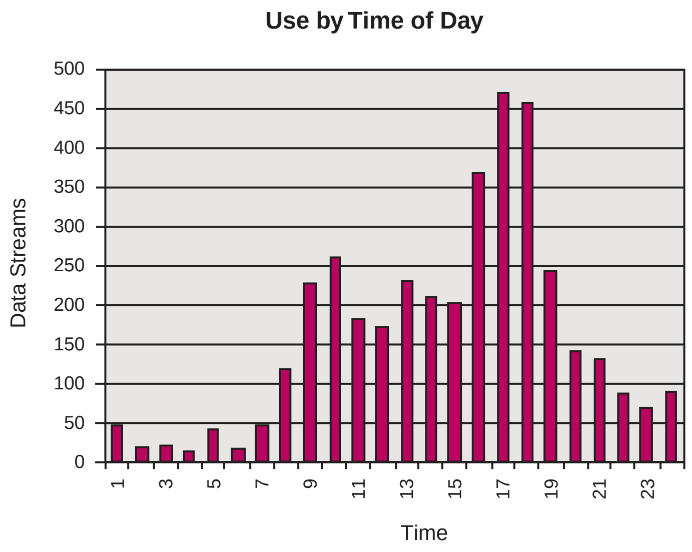


Figure 3.12: Data stream connection by time of day

Table 3.2 Internet Domains using the Busview data stream, sorted by the amount of traffic

Number	Domain
1686	.com (Commercial)
739	.edu (USA Educational)
661	[unresolved numerical addresses]
555	.net (Network)
43	.us (United States)
41	.gov (USA Government)
40	.org (Non-Profit Making Organizations)
35	.mil (USA Military)
10	.nl (Netherlands)
9	.ca (Canada)
3	.jp (Japan)
1	.no (Norway)
1	.kr (South Korea)
1	.es (Spain)
1	.au (Australia)

3.4 User Comments on Busview

Email comments on Busview were solicited on the WWW page used to launch Busview. The comments received are included in Appendix B. In general, Busview was well received, and several useful suggestions for improvements were included in the comments.

4. Conclusion

Both Busview versions were well received by users. Comments from Busview-X were used to improve the design of Busview during the SmartTrek deployment. Refinements to Busview are ongoing in response to the comments received through the SmartTrek program. Increased use is expected as additional sites on the Internet include a link to Busview. Internet visibility seems to have the greatest likelihood of increasing the number of users. In general, both applications, Busview-X and Busview, seemed to acquire a set of “hard core” users over time.

BIBLIOGRAPHY

1) Dailey, D.J, M.P. Haselkorn, and D. Meyers. "A Structured Approach to Developing Real-Time, Distributed Network Applications for ITS Deployment." *ITS Journal*, Vol. 3, No. 3, 1996, pp. 163-180.

2) D.J. Dailey, M.P. Haselkorn, K. Guiberson and P. Lin. *Automatic Transit Location System*, Washington State Transportation Center - TRAC/WSDOT, Final Technical Report WA-RD 394.1, 49 pages, February 1996.

APPENDIX A: Busview-X Comments

Status: student
Freq: Every Day
Comment1: First of all I'd like to say this is awesome! I'm quite impressed, good job so far. Let's see, I couldn't get the zoom feature to work. I would be nice if i could zoom out a little. It would be nice to see farther than the UW, just to know if my bus is coming. Remember most people can't check this at the bus stop, so we need tim to get out there.
Comment2: Yeah, the content of the map looks fine. Perhaps you could put a few more street labels though.
Comment3: I really don't see this as being "useful" so to speak, until it has a larger range. Like I said before i can't really watch my bus coming, and then get out fast enough to catch it.
Comments: As I said before I think this is really neat. I am curious as to how you are getting your data from the busses? Are you filtering raw data? or are you getting updated bus positions electronically? Also, what tools did you use to develop this? Again, great job!

Status: student
Freq: 3-5 times per week
Comment1: Great!!!!!! A bit bigger coverage would be great.
Comment2: Fine . Great. Clear.
Comment3: I just found about it. I plan to use it so I can figure out when the 48 leaves montlake so I can make it to the bus stop from my office. The 48 is always late, so this is absolutely wonderful, specially in the cold winter and rainy months.
Comments: good job. great idea.

Status: student
Freq: Every Day
Comment1: seems pretty sufficient for leaving campus -- obviously a larger range would be great.
Comment2: obviously the maps could use some work. I certainly wouldn't want to navigate by them (i.e. discerning between 15th and the ave). As long as you know the basic layout of the area, they work.
Comment3: I will try using them later tonight when I go home (which brings me to an idea). It would be excellent if you could click on a bus, then click on a location and get a guestimate of when that bus would be at that location. (even better would be the option where you tell it where you are, and it beeps to warn you to leave to catch the bus!)

Comments: the x-windows interface is a little awkward. that is, it'd be nice to have this in Java or something that was multi-platform (so I could run it from my home). but overall, this is really cool!

Status: student
Freq: 3-5 times per week
Comment1: I live at 55th St and 29th Ave NE, and would like coverage of that area. Also, if I can't have the system predict an arrival time (see below) I need a larger view to extrapolate for myself, 15-20 minutes BEFORE arrival in the U district
Comment2: It's tough to read the route numbers when they overlap with a street, on my greyscale monitor. Large view helps, as does the manual filter. I'd like a filter option to view one DIRECTION only You could eliminate seconds from the yime displays The street names don't help me and clutter the display. The only way I know which streets the names in the U district refer to is because I know the U district. Otherwise I'd be totally confused. As it is, I don't need them.
Comment3: Curiosity only. By the time I see that bus X is at the U-district, it's too late for me to get there from my office on campus. After trying 'zoom' I got an error "can't read zmx2. No such variable'
Comments: I'd prefer a system in which I could name a route and a stop, and I would receive a prediction of when the next bus(es) would leave the stop. Impressive use of web technology! It'd be nice to see the direction of movement on the FIRST display of the bus. Otherwise you will encourage people to leave busview running all day, so it will be updated when they need it. Will that impact your server?

Status: faculty
Freq: Every Day
Comment1: Map regions should be expanded north and between downtown and campus
Comment2: bus numbers and times are difficult to read through the streets on a black and white terminal.
Comment3: Mostly useful if I need to take a bus different from those I usually take.

Status: staff
Freq: 3-5 times per week
Comment1: This is really cool. But how do you track the buses?
Comment2: Pretty nice layout for as much area as you have to cover.

Status: student
Freq: 3-5 times per week
Comments: I'm getting to this survey form, but never getting busview to actually appear on my screen. Do I need to be running Netscape on the same machine that I specify as my display? Thanks.

Status:	student
Freq:	Every Day
Comments:	I don't see the map displayed on my xterm? why?

Status:	student
Freq:	Every Day
Comment1:	This is amazing...
Comment2:	Easy to read

Status:	student
Freq:	Every Day
Comment1:	this is truly fantastic and a fantastically great thing. i don't yet know whether i'll use it regularly, cause, say, i don't have X at home which is where i am in the mornings trying to figer out whether to go get this bus or wait for the next...
Comment2:	a LITTLE bit of color could go a long way. i suggest making the bus text red so that it'sa easy to pick out

Status:	student
Freq:	Every Day
Comment1:	i just sent a message, in which i suggested using color. turns out there was a problem in my X display. it looks much better now.....

Status:	student
Freq:	Every Day
Comment1:	It's not clear how to get to different parts of the city. I'm interested in seeing buses north of the current map view.
Comment2:	It's fairly easy, but you black on white would probably look better. And some indication for street names would help (probably something like clicking on a street would give the street name).
Comments:	Probably could save a lot of network bandwidth by just sending where the buses are to a local tcl client program rather than through X.

Status:	student
Freq:	Every Day
Comments:	Busview map appears for a while, no menus, no action, then the window goes away.

Status:	student
Freq:	1-2 times per week
Comments:	Unfortunately, the program just said "cannot connect to server" and refused to display anything of interest.

Status:	student
Freq:	3-5 times per week
Comment1:	The map itself is great, but the area of coverage is not. My office is about 5 minutes from the bus-stop, so I would like a much wider range of coverage so that I can see when a bus (the 48) is approaching the University from Capitol Hill.
Comment2:	The appearance of the maps are good, except for one problem. The contrast between the light green bus time and the white of the streets makes the bus time difficult to read at some points on the map.
Comment3:	Once the area of coverage is extended, I will use the system frequently to determine when to leave my office so that I can spend less time waiting at the bus stop. With the current area of coverage, the system is not very useful to me.

Status:	student
Freq:	Every Day
Comment1:	Further south of U of W would be helpful. Perhaps using a smaller map?
Comment2:	Text indicating buses hard to read. Perhaps add X or bus icon to make easier to spot and sort out from the clutter?
Comments:	Very interesting service! Hope this is expanded!

Status:	student
Freq:	Every Day
Comment1:	It's more than enough for U-Dist, but of course it will be much better to include other maps besides downtown.
Comment2:	A bit congested even for small fonts. It is horrible when I want to display the street name at the same time.

Status:	staff
Freq:	3-5 times per week
Comments:	I haven't looke at them yet --don't know if this mails off first --or will give me a chance to comment after I've browsed abit.

Status:	staff
Freq:	Every Day
Comment1:	Looks Great for Metro Riders. (I ride CT, but I enjoy watching Metro Routes move around)
Comment2:	The Busses (#'s and times), Roads, and street names are all the same color. And they tend to overlap and make it more difficult to read. Is there anyway to make different colors for different things? (it could be my Xserver running out of colors also)
Comments:	I have heard a lot about this program and this is my first experience with it. I really enjoyed this and will probably reference it more in the future. Is there a way to examine other areas than downtown and the u district? (i.e. pan the viewing area or zoom out?)

Status:	student
Freq:	Every Day
Comments:	I can't get the damn thing to work!!!!

Status:	student
Freq:	Every Day
Comment2:	The direction of the buses are difficult to see. Maybe adding color or enlarget the mark would help. Help is not available to explain what the metro logos are for on the map and what the time under each bus means.
Comment3:	Useful to help me decide when to leave. And it's fun!!
Comments:	Need help to develop a Windows version? Maybe I can help write it.

Status:	student
Freq:	Every Day
Comment2:	Very hard to see on grey scale monitor.
Comment3:	I'm not sure, it's my first time.
Comments:	On line time table is definitely good to keep.

Status:	student
Freq:	Every Day
Comment1:	it would be nice if a bus you are trying to follow is out of the screen but you are told here it is.

Status:	student
Freq:	Every Day
Comment3:	The display could not share colors with Netscape (the only other color application running) on my Sun 4/110. In monochrome, the menus did not have visible text. The map and schedule windows did not redraw in response to expose events, but did redraw if I made some menu request.

Status:	staff
Freq:	Every Day
Comment1:	Cannot tell the content of the map since it disappears so quickly.
Comment2:	Map not visable long enough to tell.
Comment3:	See responses above.
Comments:	As noted above, the screen doesn't appear long enough to see anything useful. The initial image with the map appears. Next, some asterisks appear with some writing near them appears. Instantly, after that, the screen disappears. When this problem gets fixed, please send an email and let me know and I'll try again.

Status:	staff
Freq:	3-5 times per week
Comment1:	It would be nice to see more of the city, especially NW and NE Seattle and Capital Hill.
Comment2:	The maps are quite clear

Comment3:	Very useful (and cool :-). I used it to see where my bus was.
Comments:	Are there any plans to make the software available for running on any UW Unix system?

Status:	student
Freq:	3-5 times per week
Comment1:	The maps could extend a little bit further south of Montlake, and the lines to draw the streets could maybe be kept thinner, or be given less intense colors. As it is now they sometimes make it hard to read the bus sugnals.
Comment2:	see above
Comment3:	I just used them to explore this new service, I will probably use them more often in the future to time the departure from my office.
Comments:	Sometimes a bus seems to "disappear", i.e. stop sending signals even though it's last signal was somewhere in themiddle of the map.

Status:	staff
Freq:	3-5 times per week
Comment1:	It would be nice to extend the range a bit more. The elements seem fine.
Comment2:	Use of color would be nice to key out items. I get a white on black display and it is hard read if more than a few buses are there.
Comment3:	They looks like they could be usefull for me, except that our group at work is now using magic cookies for x-term authentification, so the current system of xhost + to get in will not work without re-starting the terminal in a less secure mode. This means that I bascially could not use it as a 'real' monitoring tool. I reallyt like it otherwise, and if there were a way around this problem, I would definitely get a lot of use out of it.

Status:	staff
Freq:	3-5 times per week
Comment1:	yes
Comment2:	just fine
Comment3:	not too useful. The initial drawing I could not find route 370...So I selected only route 370 and it was nowhere to be seen. At 5:10 I think one of them should have been on the map at least...

Status:	student
Freq:	3-5 times per week
Comments:	The color map does not work really well when I am also using Netscape

Status:	student
Freq:	3-5 times per week
Comment1:	the maps are great.
Comment2:	Put the bus info in a different color. like red It is currently VERY difficult to read.
Comment3:	they are a great tool

Comments:	who are you guys located? ivhs? what department is that?

Status:	staff
Freq:	Every Day
Comments:	I was never able to access your site due to my inability to find the "session window". Your instructions were not very clear.

Status:	student
Freq:	3-5 times per week
Comments:	I would like a way of marking up a certain bus so that something beeps on my terminal when my bus is within range. For example, when the 48 hits NE Pacific I would like to get a beep.

Status:	student
Freq:	Every Day
Comment1:	Yes, everything is great! Now, all the colors are working! Please read the comments.
Comments:	I've another question. I'd like to automate the opening of the busview in a UNIX script, but therefore I need some information about what I have to send to which computer. Maybe you have a special port, to which I can send my name and ID-number and Display, and the busview will automatically started. It's a little bit boring to enter all this stuff everytime in Netscape and loading Netscape takes so long! I'd really appreciate some info, because your busview is used a lot here in our lab!

Status:	staff
Freq:	Less than one time per week
Comment1:	Maps are OK. Much more accessable
Comment2:	On Screen OK (Monochrome monitor). Still waiting for the print out.
Comment3:	As good as the kiosk maps.
Comments:	Myn kids fly metro, this may help me help them. We have Mac at home, not X-term so use is limited.

Status:	student
Freq:	Every Day
Comment1:	It might be neat to extend further south (since I live in the south end). I really like the idea of both the downtown and the U-dist maps.
Comment2:	The map is very easy to understand and follow.
Comments:	I think this is a really neat, clever, and wonderful idea. It might be nice to format the schedules a little better, though.

Status:	student
Freq:	Every Day
Comment1:	The two maps are nice. It doesn't look like they are adjcent, though. Could that be changed?
Comment2:	The bus numbers are difficult to read when the map is viewed in full size. How about a menu option that toggles opaque or transparent text? That is, I could optionally make a block of text overwrite the map background, obscuring the underlying streets.
Comment3:	This is my first use of the system, so it's more for my personal enjoyment than anything. In the future, what I'd probably do is figure out (1) how long it takes to walk to the bus stop, (2) how far my bus can travel in that time, and then (3) work backwards when I must leave in order to just catch the bus; I hate waiting for the bus.
Comments:	I'd like to be able to open more than one map at a time. Is this possible? Also, what about seeding the bus locations when the system first starts, instead of opening an empty map? Great system!

Status:	student
Freq:	Every Day
Comment1:	greater resolution would be good...many students are not from the area !
Comment2:	what is there is OK for itself
Comment3:	it didn't work. The buslines should be displayed on a smaller scale map.

Status:	student
Freq:	3-5 times per week
Comment1:	I live on Beacon Hill. I didn't see an option for viewing the South end.
Comment2:	The maps were too cluttered. I think that a less detailed map should be drawn by default. The user can then use the zoom function to get greater detail. Note that you don't have a zoom out function. Though I figured out that the original view could be restored by reselecting which map I wanted to see, a zoom out or restore original view should be listed next to the zoom choice on the menus.
Comment3:	The maps were pretty slow to come up. Maybe a less detailed map could be used to setup a zoomed view. TRhat might speed things up a little, though I don't know what's going on at your end.
Comments:	Why not have this accessable through the Web? Make it work like the WS DOT traffic maps. As for the functions, those can be put in a form that the user fills in. Sure it's not real-time, but it's real close.

Status:	student
Freq:	3-5 times per week
Comment1:	Nothing came up. The screen flashed and I was dropped into the survey screen...

Status:	student
Freq:	3-5 times per week
Comment1:	I got a message about insufficient colors and ended up with a black and white map that was pretty useless--even with the large ofnts option it was almost impossible to make out buses and route information.

Status:	staff
Freq:	Every Day
Comment1:	Fairly sufficient.
Comment2:	The zooming mechanism could be improved on.

Status:	staff
Freq:	Every Day
Comments:	No buses appear on the map and attempting to use the bus filter results in the following message: Error: font "-*-helvetica-medium-r-normal-*-12-*-*-*-p-*iso8859-1" doesn't exist The map's colors appear good however no bus numbers appear nor times. Perhaps related to the font problem noted above.

Status:	faculty
Freq:	1-2 times per week
Comment1:	sufficient
Comment2:	Appearance is fine
Comment3:	Just checking them out, but the map will be very useful in the future.
Comments:	Unbelievable!!

Status:	faculty
Freq:	Less than one time per week
Comment1:	I see no maps, and it is not clear how to bring them up.

Status:	student
Freq:	3-5 times per week
Comment1:	More map areas would be good.
Comment2:	The maps appear OK.

Status:	staff
Freq:	Less than one time per week
Comment1:	Did not get a route map when selected it. (Crossroads)
Comment2:	Very nice
Comment3:	May use maps for patient information. I think they will be very heplpful.

Status: s	tudent
Freq:	1-2 times per week
Comment1:	The maps are well done, but it would be nice to have a larger range of areas, such as Green Lake, Ballard, etc. It would also be nice to have info avaible in data form, so that users or devolopers can customize use. What I mean is, if I

take the 48 every day, I might want to know when it reaches 50th street, since I get on at 45th. By using the data, rather than the image, I could write a program (or use a pre-devoloped one) to sound a buzzer or something when this event occurs.

Status: staff
Freq: Every Day
Comment1: yes
Comment2: often hard to read bus number, even if large; often hard to tell direction of travel. Tried it on a 1-bit VaxStatation -- displayed prety well. Tried it on an SGI -- the color helped. Tried it on a color NCD xterm -- got *no* bus info?!
Comment3: 1. Very neat. 2. I commute from the U to Kirkland; take the 260. I was able to see when it left the downtown area an estimate when it would get to Montlake. It was 15 min. late yesterday and I could wait/work in my office instead of wait at the bus stop.
Comments: How about a text version -- in a schedule format -- either in place of a schedule or have actual time next to schedule times. on the web, of course

Status: student
Freq: Every Day
Comments: Menus on top don't respond to mouse -- but they do respond to keypresses. This is on a SGI IRIX 5.2. This makes it impossible not to look at all the routes. Really I'm only interested in one bus...

Status: staff
Freq: Every Day
Comment1: Looks like it could be very useful for checking on when to leave for my bus! Will the map show tunnel buses in the tunnels?
Comment2: What is the significance (if any) of color?

Status: staff
Freq: Every Day
Comments: Could there be a warning (bell, popup, whatever) when a particular bus leaves a particular stop? This would allow me to, for instance, work up to the last second, and not have to worry about missing my bus, and not having to wait for a very late bus.

Status: student
Freq: Every Day
Comment2: It is difficult (sometimes impossible) to see when there are more than one bus stop at one place and the texts overlap each other.

Status:	student
Freq:	Every Day
Comment1:	I'd like the area South of Montlake to be larger so that buses approaching from that side can be seen earlier (eg: #25).
Comment2:	I use a monochrome Xterm, so the small lettering is hard to see. Perhaps using a highlight color would help.

Status:	student
Freq:	1-2 times per week
Comment1:	The map coverage is adequate.
Comment2:	The maps are actually too detailed. They illustrate streets on which buses never run. For those who are not from Seattle, I think that these maps may b confusing. Maybe an option for a map with only 45th, 15th, the ave, roosevelt... and other major streets would help.
Comment3:	The N_S orientation is ok. For some, that not the way they see the U-dist. Could there be orientation options?
Comments:	I didn't see any buses on the maps.. Is the prog working or did I do something wrong. On this page. Maybe some steps telling what options should be set in order to see a common route. 44?

Status:	student
Freq:	Every Day
Comment1:	The map coverage is very useful for the areas that are most used.
Comment2:	Everything is very small. It will be difficult for people with smaller monitors. I think it's necessary though to get all the information though. The zoom feature works well and fixes that problem.
Comment3:	I might use them to get bus schedules and see when my bus was coming.
Comments:	It might be useful to overlay a route onto the map for a given bus. That way you could see where it takes you since not all the street names are on the map yet. Also, an alarm feature that would let you know when a particular bus reaches a certain point would be useful for people that want to work up to the last minute, but not miss the bus. :)

Freq:	Less than one time per week
Comment1:	Nice map

Status:	student
Freq:	3-5 times per week
Comment1:	Would like to see more coverage.
Comment2:	Use of color would help discriminate buses from background. Other use of color, e.g. types of streets. (Appeared only black & white)
Comment3:	Not used, but could indicate whether one's missing a bus or nor :-)
Comments:	Looks promising.

Status:	student
Freq:	Every Day
Comment1:	Should contain all possible routes serviced by meto and community transit.
Comment2:	More detail

Status:	staff
Freq:	Every Day
Comment1:	I don't know how to run this
Comment2:	I don't see any map
Comment3:	this is useless

Status:	student
Freq:	Less than one time per week
Comment1:	The content is fine
Comment2:	It is nice and clear
Comment3:	I was just curious because I stubled across the maps.
Comments:	It is *way* too difficult to bring the maps up to actually be of use. To get a bus map, you would have to go to a campus computing center, sign out an X-term, log in, launch netscape, find the page (or have it bookmarked), add the xhost, type in your ID information, and click OK on the legal mumbo-jumbo. It's just not worth the effort to bring up a map of where the busses are! It *would* be cool to have it displayed at the bus stops on campus, though. I'm not sure if there is any way to do so without risking vandalism to the displays though.

Status:	staff
Freq:	1-2 times per week
Comment1:	The coverage is adequate
Comment2:	Perhaps you should remove or dash streets where buses don't travel
Comment3:	They're useful. They helped me select a bus to use and when I should leave my office. I will use it in the future.
Comments:	When two buses having the same number are on the same street but traveling in opposite directions, perhaps they should be a different color, as well as havin the directionsl arrows.

Status:	staff
Freq:	3-5 times per week
Comment1:	The coverage is fine for catching a bus from the U-district, but won't help for coming in to work from the Eastside in the morning.
Comment2:	Sometimes I couldn't see the direction arrows. Otherwise they were adequate.
Comment3:	I may begin using the maps to decide when to actually leave my office and walk to my bus stop. Also it will be helpful to know if I have missed my bus and should wait for the next one. This will provide up to 30 minutes of additional work time. If I walk to the bus stop and then find out I have missed the bus, I don't have time to return to work and accomplish anything useful. But if I know before I leave that I am going to miss the bus, then I can continue

	to work until the next one comes. Also, I can phone my wife to let her know I will be late.
Comments:	The route schedule for the 167 should have come up wider on my screen or with a smaller font so that I could see the whole route listing in one line.

Status:	faculty
Freq:	Less than one time per week
Comment1:	I like the bus filter. It's too busy otherwise.
Comment2:	The windows-like interface is dumb. Motif style buttons would be preferable.
Comment3:	I'll be using them to see when I should catch a bus. Thanks

Status:	student
Freq:	1-2 times per week
Comment1:	It's too bad Bellevue is left off, but I'm sure that would have made the program twice as hard. Seattle and UW areas are covered well.
Comment2:	The bus text is hard to read because the white streets obstruct the light green bus text. The bus text is the most important, and should be the dominant color. The street names show in the same white as the street, so they are even harder to read. If I had it my way, the background would be white, not black. The street would be a 50% grey, good enough to see but not obstructive. The buses and street names should have unique and dominant colors, like blue and dark green.
Comment3:	The interface to showing street names is a bit clugey. First it brings up a seperate window. It should just change the cursor. You can't remove streets names that you already showed. You have to reset the entire map I guess.
Comments:	The way this program is set up, a person must have an X Server (which is not common), and they cannot have their preferences saved. I think a better way to go would be to write a Java front-end that accesses your database. Anyone in the world could see the maps, zoom in on them, whatever. If they wanted to see "live" buses, they would have to enter the student number, name, and click 'OK' to accepting the terms. The Java program could also save their preferences, so they could have their own selection of bus filters, default map choice, and even color selection!

Status:	student
Freq:	Less than one time per week
Comment1:	Whenever I open an extra box, the color map goes to monochrome
Comment2:	Whenever I open an extra box, the color map goes to monochrome

Status:	student
Freq:	Less than one time per week
Comment1:	Seems OK.
Comment2:	Everything looks quite nice.

Comment3:	I was curious to see if the service had changed any since I last saw this service, over 6 months ago. It's kinda neat that it works, but not too useful since it takes so long to bring up the service. (By the time you found your bus, it would have stopped at your pickup point, and left again!!)
Comments:	This doesn't seem very useful, but it sure is cool!!

Status:	student
Freq:	3-5 times per week
Comment1:	More extensive coverage would be nice, if possible.
Comment2:	The overall appearance of the maps is good. I like the feature of clicking on a bus number and having the scheduke pop up.
Comment3:	I think BusView is a wonderful program! I always look at the map before catching the bus at night. I now have it timed so I can be at the bus stop just before the bus arrives.
Comments:	I wish the program was working today, Nov. 19, 1996, because the snowstorm has made the bus schedule irrelevant. It would be really nice to know where the buses are in real time.

Status:	student
Freq:	Every Day
Comment1:	Where is it?

Status:	student
Freq:	Every Day
Comment1:	It would be neat if a map extending South of Seattle could be provided. It may already, but I simply couldn't access it (BusView is slow over a 28.8Kmodem!).
Comment2:	Sure!
Comments:	What would be _really_ neat is if the BusView client could be downloaded and run _locally_ on my X-windows machine, and all that's sent over the network are the bus position updates. That way, the client is pretty snappy (running locally), but still requires access to roadway.ivhs for the bus data.

Status:	student
Freq:	Every Day
Comment1:	perfect
Comment2:	yep
Comment3:	quite useful

Status:	staff
Freq:	3-5 times per week
Comment1:	It would be helpfull to have city wide maps.
Comment2:	There should be a way to eliminate the street names after adding them.
Comments:	Good work!!

Status:	student
Freq:	3-5 times per week
Comment1:	Around the campus, the coverage is sufficient. Of course, since I live away from campus, I'd like coverage improved off campus!
Comment2:	The elements are fairly easy to identify, mainly because I know the exact lay of the land. If I didn't, I'd like to have more landmarks, and a more accurate pointer to each of the landmarks (rather than the large, relatively ambiguous labels currently used).
Comment3:	I was just checking the map out; just learnt about it. I haven't used it yet.
Comments:	First off, I think this is a great idea; hope you guys can push it through to something widely used! The following additional features may be useful: 1)Arrows showing which way the bus is moving (because most buses ply in both directions simultaneously on a given road). 2)More frequent updates: this is because I most need the service when I think I may have missed the bus (or have very little time to catch it). Since the bus-stop is about 30 secs from my office (it's probably so for many people on campus), I'd like a resolution of around 30 seconds at least.

Status:	student
Freq:	Every Day
Comment1:	Richmond Beach, Richmond Highlands, Shoreline are areas in which I would be interested in seeing.
Comment2:	It would be helpful to identify in which directions the buses are moving, special conditions on particular buses (e.g., is my bus dead?), highlight routes of interest, and center the map on buses of routes of interest. Some indication of whether the bus is approximately on schedule would be useful, too. Estimated times of arrival at selected stops?
Comment3:	I think it's pretty cool, but... It doesn't seem very useful. If your bus doesn't appear, you don't know why. Also, the interval between updates is too great. Finally, unless the maps are displayed on consoles at major bus stations, nobody will have access to the information when they need it. What information do people need? - what route goes to my destination and how long will it take? - where is the nearest stop for that route? - when will the next bus reach that stop? - (once the person reaches the stop) did I miss the bus or not? if I did, do any other routes take me where I want to go? which of the above questions do the maps address?
Comments:	Shouldn't this be a Java applet? That might make it easier to integrate with Metro's web page, allow people to use it from home, etc. Could you superimpose current-traffic data from the DOT web site onto your maps?

Status:	staff
Freq:	Every Day
Comment1:	I have never used it before, and currently all I can is this survey form.

Status:	staff
Freq:	Every Day
Comment1:	Very nice
Comments:	I would like to be able to set "permanent" bus filters. That is, I don't want to have to constantly add the buses I am interested in watching one at a time.

APPENDIX B: Busview Comments

**
Date: Thu, 07 Jan 1999 16:34:37 -0800
From: Stefan Kramer <SKramer@cac.washington.edu>
Subject: Does the Busview service exclude Community Transit buses? I didn't see any statement at http://www.its.washington.edu/busviewplus/ that Busview only shows METRO buses, but after having looked at the map for the UW campus area for a while, it seems like it. Otherwise, this is very impressive!

**
Date: Mon, 25 Jan 1999 10:59:04 -0800
From: "Mark Falge" <mark@phos.com>
Subject: Busview!
Wow! Busview has really come a long way since I last took a look! Looks like this is going to become a very valuable tool for me. I'm sure you're busy with many suggestions but I had an idea that I know would at least be very useful to me. Would it be possible in the future to set up something similar to a proximity alert for buses?
Thanks for all of the hard work.
Mark

**
Date: Tue, 26 Jan 1999 17:13:21 -0800
From: "Victoria H. Brophy" <torih@mendel.genetics.washington.edu>
Subject: Busview suggestions and questions
I recently discovered busview and I really like it. Does every Metro bus send a signal for busview or are some not shown on the map? There are two features I would find useful. One is the ability to scroll the map. The other is a find feature. I ride bus 205 and I'd like to know where on its route it is, but "route progress" doesn't help me because there's only one bus at a time. I have to find it before I
can ask where it is.
Thanks,
Tori Brophy
University of Washington

**
Date: Wed, 10 Feb 1999 19:48:48 -0800
To: busviewplus@its.washington.edu
From: Don Sheythe <donster@wport.com>
Subject: Busview - locating bus on map is difficult
Dear busview team,
I think busview is really cool. It has greatly improved since the version that I saw a few months ago.

**
Date: Sat, 13 Feb 1999 08:04:03 -0800
From: Jim & Debbie Weiss <jdweiss@gte.net>
Subject: BusView
This is WAY cool. Please please please, get CT on here, you could make my commute life so much better. Great idea, great job, and fantastic use of technology.
Thanx
-jww

**
From: “Sylvia French (ApRoberts) (Exchange)” <sylviaf@exchange.microsoft.com>
To: Seattle Web Grrls List <swg-list@drizzle.com>
Subject: RE: (discuss) see where your bus is
Date: Tue, 16 Feb 1999 15:13:54 -0800

This is the BEST applet EVER!!!!

>——Original Message——
>From: Anne Baker [mailto:anneb@real.com]
>Sent: Tuesday, February 16, 1999 3:09 PM
>To: swg-list@drizzle.com; nwr-list@nwlink.com
>Subject: NWR: (discuss) see where your bus is
>Here’s a very cool site that allows you to actually see, in real time,
>where the buses are. You need NS 4.5 or better or IE 4.01 or better.
>
>http://www.its.washington.edu/busviewplus/bus_launch.html
>
>Enjoy!

**
From: Jones, Steve [SMTP:SJones@rwbeck.com]
Sent: Tuesday, February 23, 1999 2:51 PM
To: ‘Michael J. West’
Subject: testimonial & suggestion box
With Busview, I was able to select the route and “watch” on my PC to see when my bus was actually starting its route. When I saw the bus enter the route about 8 blocks from my office, I had just enough
time to leave my building and go to the bus stop. The bus ended up being about 20 minutes late that day, but with Busview, I was able to spend the wait time in my office catching up on work instead of
waiting out in the cold.

**
From: “Glenna Johnson” <glennaj@earthlink.net>
Subject: Re: Fwd: Busview no longer works
Date: Tue, 2 Mar 1999 09:39:32 -0800
Thanks for the prompt response! It works again! Also, I wanted to let you know that I love this applet. I was wondering, though, if there were any plans to set up some type of outbound phone system to “warn” people that their bus was a given distance or number of minutes away?
Thanks again for your help!

**
Subject: BusView feature suggestion
From: Tessa Lau <tlau@cs.washington.edu>
Date: Wed, 03 Mar 1999 11:09:34 -0800
Hi there,
I’m very impressed by the current BusView applet. I’ve tried it in the past, and you have since made vast improvements in the user interface and stability of the applet. Thanks for a great service! I’m glad to see it improving.

**
Date: Wed, 03 Mar 1999 10:44:18 -0800
From: Don Sheythe <donster@wport.com>
Subject: Busview Comments - You guys are soooo close!
You are really close to something I would use if it were easier/more convenient to use.
PS I can see that you guys have put a lot of hard work into this project. It’s really cool.

**
From: “David and Kaylin” <kingneil@uswest.net>
Subject: Your site is brilliant!!
Date: Tue, 27 Apr 1999 07:21:19 -0700
Someone just sent me an e-mail with a link to your site. After 5 minutes of playing around with it I can say only one thing. Freaking Genius!! Excellent work. This is by far one of the most useful sites I’ve ever seen. Way to go guys. Keep it up.
David Neil

**
Date: Mon, 22 Mar 1999 09:55:33 -0800
From: “Mark Falge” <mark@phos.com>
Subject: Bus View
Looks like Busview is really coming along...thanks for the program. I see that the perimeter alarm on the ‘Route Progress’ bar is going to be added soon...I can’t seem to get it to work yet. I was also wondering if it would be possible, at some point, to set up preferences for Busview...for example when I launch the application the specific map for my area would be loaded and my perimeter alarms would be set. Thanks again for the program it’s really proving very useful these days!

**
From: “Mark Falge” <mark@phos.com>
Subject: RE: Fwd: Bus View
Date: Mon, 22 Mar 1999 11:12:24 -0800
Thanks! That did the trick. This program is getting better and better... You have no idea how much your program has helped in alleviating the frustration my girlfriend and I have in missed busses! More people should know about this. Thanks again Joel.

**
Date: Thu, 25 Mar 1999 00:45:23 -0500 (EST)
From: Miles Erickson <miles@exchange.cc>
X-Sender: miles@theta.pair.com
Subject: Future of BusView?
Are there any plans to migrate BusView to a stand-alone Java application that, while still being OS-independent, would allow the flexibility to do such things more efficiently? Just curious.. thanks for your excellent work!

**
Date: Sat, 27 Mar 1999 11:48:56 -0800
From: Don Sheythe <donster@wport.com>
Subject: Busview Comments - Cool! I like the new route selection!
Dear Busview team,
I have seen the new features you’ve added to busview and I am very impressed. It makes it so much easier to use! It used to take me about 5 minutes to get the bus I wanted, now it takes about 10 seconds! The only thing that confused me was that the timepoints listed in the ride selection window were not in sequential order along the route. (for example 32nd and 143rd were listed below 35th and 125th on the 65 route into the UW) However this is a minor complaint on a major improvement!
Now if busview could save the state of the open windows to some cookies so I wouldn’t have to bring up the same route and set the same alarm each time I launch it 8-)

Thanks a bunch, I think I’ll be using Busview a lot more now!
Don

**
Date: Wed, 31 Mar 1999 13:43:50 -0800
From: Andrew Williams <anwilli@rei.com>
Subject: BusView
Great idea putting this on the net! My bus stop is uncovered and it’s great to know when the bus is actually coming. I just wish I had found this at the beginning of the rainy season rather than near the end! I’ll just hang out in my nice warm office instead of freezing waiting for the bus.
Thank you,
Drew Williams
REI Adventures

**
Date: Wed, 31 Mar 1999 13:44:27 -0800
From: Gary Lewis <glewis@mail.prognet.com>
Subject: a couple of suggestions for busview
Hello,
Again (I have sent prior messages), I want to commend you on a great application. I love the addition of the alarm on the bus progress applet. Thanks for the opportunity to give you feedback and keep up the great work.

**
Date: Thu, 01 Apr 1999 19:17:04 -0800
From: Don Sheythe <donster@wport.com>
Subject: Re: Busview Comments - the case of the missing route 65 8-)
I can imagine what a tangled web of route information that you must have to sort through. Thanks for all your hard work, this is really cool stuff! It saves me having to wait out in the cold (a couple of nights this week have really been cold!) I cant wait for the latest version to be posted! 8-)
Don

**
Date: Thu, 08 Apr 1999 16:32:57 -0700
From: Chris Maly <chris.maly@cimedia.com>
Subject: Fwd: Re: transit resources
Thanks for the tip. How are you doing this? GPS tracking? Very cool. Is this information public domain? Could we integrate it into SeattleInsider.com?

**
From: "Swick, Reid" <Reid.Swick@METROKC.GOV>
Subject: BusView
Date: Thu, 15 Apr 1999 13:51:29 -0700
This thing is really cool!
Is there a way to preserve your Map and filter between invocations of the program?

**
From: Chris Arthur MacGregor <csa@bsquare.com>
Subject: busview comments/problems
Date: Thu, 15 Apr 1999 14:46:19 -0700
Busview is veeeeeeeeeeery cool. I just wish I'd known about it sooner - it could have saved me a lot of waiting in the rain this past winter.

**
Date: Fri, 16 Apr 1999 10:28:55 -0700
From: "MAGER,GARY (HP-Seattle,ex1)" <gary_mager@am.exch.hp.com>
I just found your Busview project on the Internet. It looks very interesting. Can you tell me how you track the buses? Does each bus have a GPS device? Thanks, -gary

**
Sent: Friday, April 16, 1999 3:31 PM
From: R. Maney
Subject: Busview
Hello and congratulations on a great application! My life is much easier since I discovered Busview. Thanks for your help,
Todd Maney
University of Washington
 Department of Physiology and Biophysics

APPENDIX C: 1994 World Congress Paper

Demonstration of an Advanced Public Transportation System in the Context of an IVHS Regional Architecture

D.J. Dailey, M.P. Haselkorn

University of Washington

Seattle, Washington U.S.A.

1. INTRODUCTION

In previous papers,[1] we have presented an overview of an architecture for a regional IVHS network. Numerous benefits were associated with the IVHS development strategy presented in those papers, including:

1) easy sharing of interagency and multi-jurisdictional data without disruption of existing operations.
2) support of existing investment in IVHS technology and system development.
3) easy expansion of sensor technologies and user applications in a straightforward, principled manner.
4) encouragement of future innovation.
5) interoperability among local, regional, and national IVHS development efforts.

This paper describes a concrete demonstration of this architectural concept - a prototype Advanced Public Transportation System (APTS) that displays on a single GIS screen (1) real-time bus locations from an AVL system owned and operated by Seattle Metro, the seventh largest transit company in the U.S.A., and (2) real-time freeway congestion data from

[1]"A Conceptual Framework for IVHS System Development," *Proceedings of the IVHS America 1993 Annual Meeting*, pp 1-7; "Traffic Information and Management in a Geographically Distributed Computing Environment," *Proceedings of Pacific Rim TransTech Conference/ASCE Third International Conference on Applications of Advanced Technologies in Transportation Engineering*, Seattle, WA.: ASCE, pp. 159-165.

inductance loops, obtained from the Traffic Systems Management Center (TSMC) operated by the Washington State Department of Transportation (WSDOT). We call this APTS BusView.

We first describe the current implementation of BusView within the context of Washington State's IVHS regional architecture, and then present both the realization of the benefits predicted in our earlier papers as well as issues encountered during the development of the prototype system.

2. BACKGROUND AND ARCHITECTURE

In this section we present as background the architecture for Washington State's expanding IVHS regional network. This geographically distributed traffic management and information system is the environment within which our real-time transit location and congestion application lives.

The central thrust of Washington State's regional network is to provide a fertile and efficient environment for the development and deployment of IVHS services. This environment is based on a client-server relationship between consumers of traffic information and traffic information sources. Specifically, the architecture provides: (1) an efficient mechanism for scaling individual developments to a regional deployment, (2) the ability to flexibly support various sensor and application technologies, (3) easy extendibility to new clients and to new or improved data sources, (4) a guarantee that existing agencies retain autonomous use of the sources they presently control while gaining access to information previously unavailable to them, and (5) support for collaborative development and deployment at geographically and politically distributed sites.

Washington State's regional IVHS network is a distributed computing environment that can best be viewed from two perspectives: (1) information flow and (2) communication architecture. From the information flow perspective, our environment can be divided into three principle types of systems: (1) data source systems, (2) data fusion systems, and (3) information delivery/display systems. Data source systems include roadway sensors and the devices required to place sensor data on the network, as well as more static sources such as map databases. Data fusion systems include devices that gather and merge similar types of data, as well as devices that combine different types of data in useful ways. Information delivery/ display systems include the applications which provide IVHS services and the devices required to place these applications on the network.

As seen from this information flow perspective, BusView consists of an odometery based AVL system (operated by Seattle Metro the transit carrier), an AVL instance server which puts the AVL odometery data on the regional backbone, an AVL positioning server which converts the AVL odometery data to latitude and longitude pairs, and a GIS application that displays, in real time, bus locations on a digital map (see Figure 1).

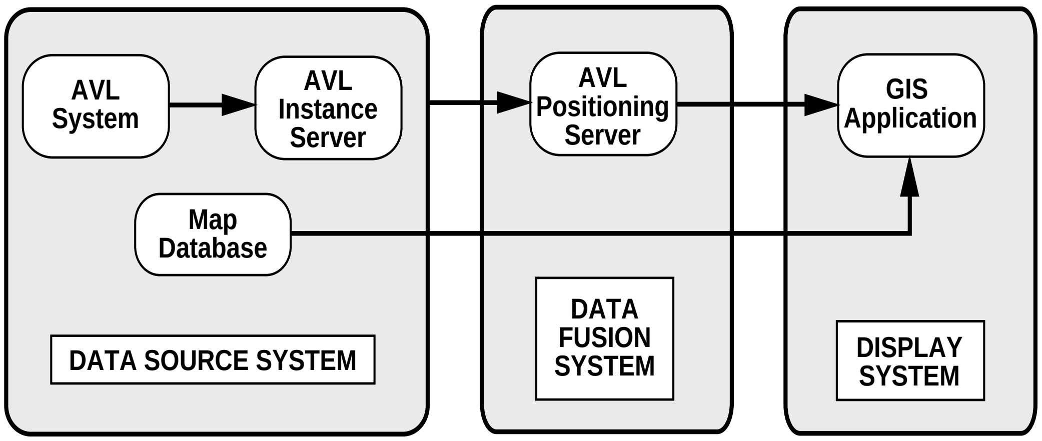


Figure 1. Information flow in BusView.

Another perspective from which to view BusView is as an instantiation of a communication architecture consisting of components that rely on distributed computing abstractions. The principal abstractions are: (a) a communications backbone, (b) a peer to peer communication paradigm, (c) a client server transaction model, and (d) a distributed name space. The communications backbone provides connectivity among all components; the peer to peer paradigm assures common communication while supporting geographic independence; the client server model provides data security and agency autonomy within the common communication model; and the distributed name space provides a mechanism for locating any system component.

This perspective emphasizes the nature of the communication within our APTS transit application (compare Figure 2 and Figure 1). In addition, this perspective emphasizes the

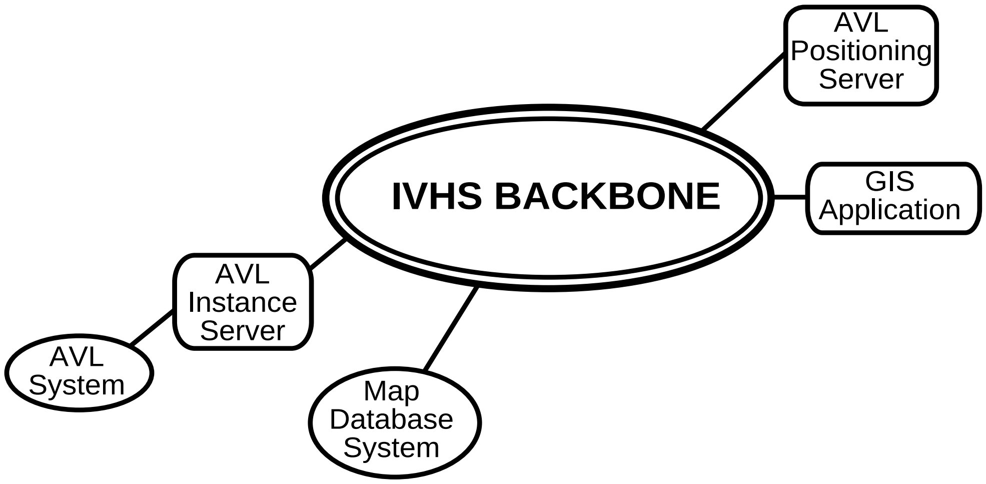


Figure 2. Communication architecture in BusView.

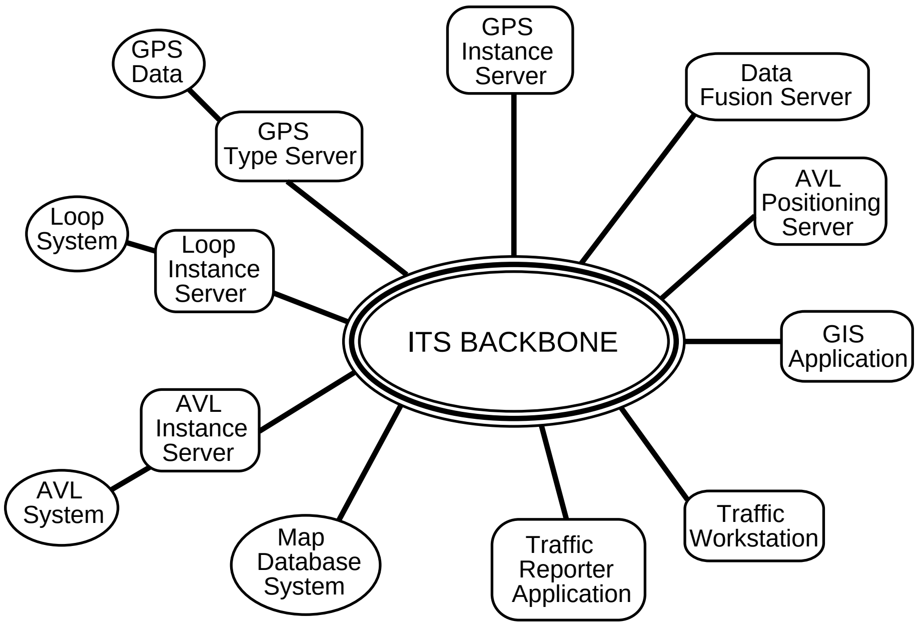


Figure 3. Overall IVHS communication architecture.

potential for similar communication with the other components in our architecture (compare Figure 2 and Figure 3). The protocols and interfaces that create the peer to peer relationships are such that a new application can communicate directly with an existing component for its own purposes. For example, a new application could communicate with BusView's AVL instance server to obtain odometery information for the purpose of tracking bus maintenance needs, rather than providing latitude and longitude coordinates (as BusView's positioning server does). Thus in this framework, a clear paradigm exists for adding new components, and new IVHS applications are designed by selecting components from the communication architecture (Figure 3) and defining the information flow (as in Figure 1).

3. HOW BusView WORKS

The BusView application is a good example of the development strategy described above. We obtain real time transit vehicle locations by interfacing with an existing automatic vehicle location system (AVL) owned and operated by Seattle METRO, and we obtain freeway congestion information by interfacing with an existing loop data system owned and operated by the Washington State Department of Transportation (WSDOT). METRO's AVL system consists of several components including a data acquisition control system (DACS) and a set of GIS command and control consoles. The data acquisition system polls each of the coaches to obtain an odometery based distance along planned routes. The DACS then broadcasts, on an Ethernet, the bus route number, vehicle ID and distance along one of 3000 patterns from which the routes are constructed. The GIS/Display system receives information about the routes assigned to each console operator and provides a graphical display. It is at the juncture of the DACS and the METRO GIS/Display system that we have placed an instance server which eavesdrops on the network and obtains the information about each of the coaches. (See Figure 4.)

METRO's AVL system is extremely complicated, but in Figures 1, 2, & 3 it is represented only by a box labeled "AVL System." This is because from the point of view of the IVHS architecture it simply represents another data source and is connected to the network in the same principled manner as a single GPS receiver would be connected. The AVL data is tapped by attaching the AVL Instance Server to the Ethernet interconnecting the components of the AVL system and in effect eavesdropping on the odometery information. In this way, METRO's information is made available to approved participants on the network without impacting METRO's use of the data. In return, METRO gains access to the data of others (e.g. WSDOT loop data) as well as to various network services and applications .

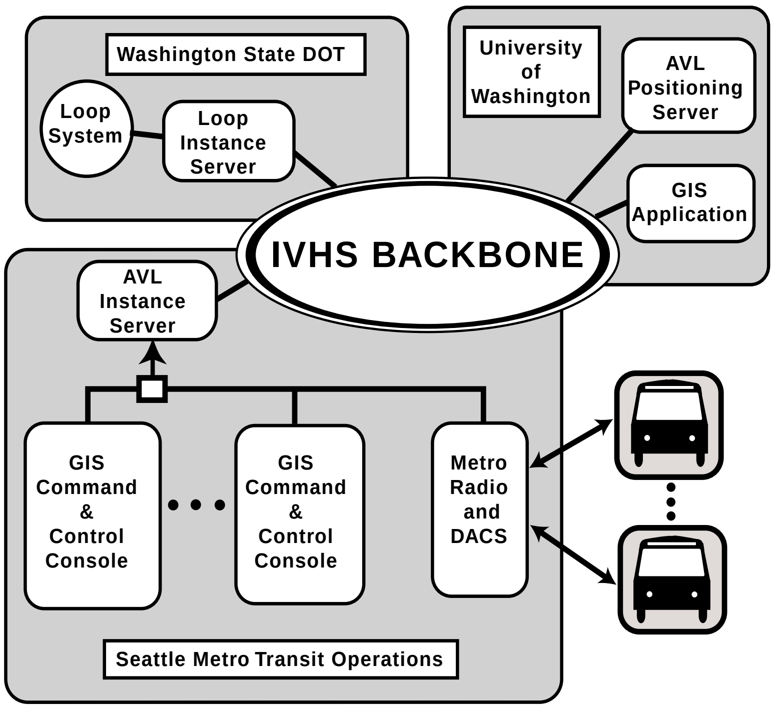


Figure 4. Implementation of BusView.

One class of network services are called fusion servers and the type server is an example of this generic category. Once the instance server has placed the AVL information on the network, it is reencapsulated and sent to a type server process running on a computer at the University of Washington (UW). The type server performs several functions:

1) it receives the coach data and, through a series of coordinate transformations and extended data structure searches, translates the location of the coach from the distance along a route pattern (a

proprietary system) to latitude and longitude coordinates suitable for comparison and display on a GIS system,

2) it provides a connection service that negotiates the addition of the client to the list of clients presently receiving the coach information, and
3) it transfers the coach data to each of the clients as the real time data becomes available.

All of these communication tasks take place over Internet style interprocess communication ports. The AVL data fusion server (physically located at the UW) is shown in the upper right of Figures 2 & 3 and called "positioning server." This component acts as the fusion and redistribution point for the AVL positioning data. It initializes a connection to the AVL instance server and requests the real time odometery information. This odometery information is combined with the routing information to produce a latitude and longitude position value for each of the buses as they are polled. The position information for each of the coaches is then available to clients for a number of uses. The first of these is a map display of real time bus locations.

To present the probe vehicle location information in a useful way, our implementation includes a traffic management type display of traffic information. In our example implementation a locally developed GIS display application accesses a set of map data derived from TIGER files, the USGS 1:100,000 scale digital maps (augmented by local measurements.) These maps can be displayed on Xterminals located anywhere on the backbone. These maps are used to give context to the probe vehicle information just developed. Access to vehicle location data, vehicle information (implemented as a network data base), and congestion data (implemented by the Loop Instance Server) is handled in the same client/server way used to establish communication between the AVL fusion server and instance server. The GIS application (BusView) makes a request for data from the server and then uses this data to build the display.

The demonstration display consists of the street network displayed on an Xterminal (the traffic workstation in Figure 3), along with the real time location of any probe vehicles and real time freeway congestion data from WSDOT. The user of this display can make selections to tailor the information. The maps can be scaled from a regional presentation to the individual street level and various layers of the cartographic information can be turned on or off (e.g. political boundaries, streams, lakes, streets, arterials and freeways). In future implementations, information about each probe vehicle will be obtained by pointing to the icon representing that vehicle and clicking a mouse button. Figure 5 is a snapshot of the current map display. The transit vehicle locations are represented by the route number placed on the map.

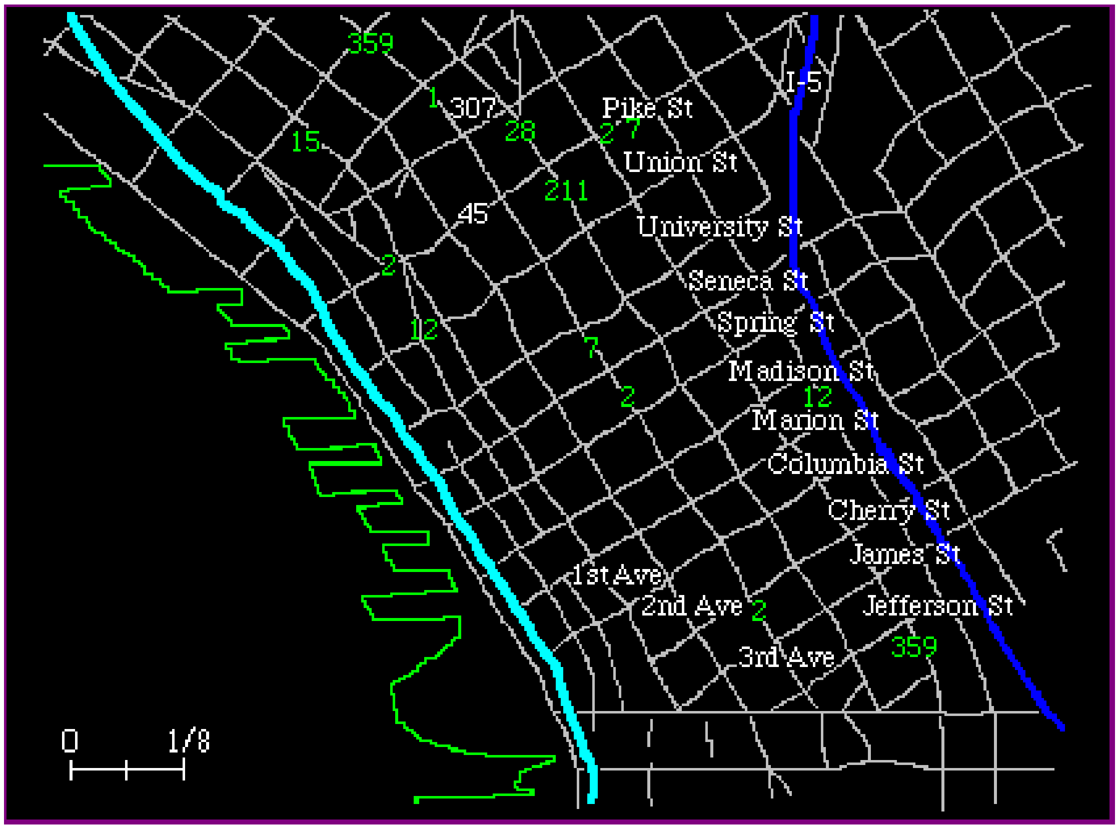


Figure 5. Display of Real Time Bus Locations

4. INSTITUTIONAL ISSUES

In addition to providing a rich development environment, the architectural framework described above also addresses institutional issues that in the IVHS arena often overshadow technical issues. Independent agencies are unwilling to share data for joint development if it means they must adjust their current procedures or sacrifice the security and integrity of their data. Our IVHS architecture is a political as well as a communication entity, since it encourages regional cooperation of transportation agencies through a principle of "enlightened self-interest." In other words, the cost of making data available on the network is minimal; the benefits are considerable.

4.1 Benefits

In the case described here, we were able to make transit data from Seattle METRO and traffic congestion information from WSDOT available on a wide area network without impacting the operations, philosophies, or integrity of the agencies whose mission it is to gather this information. For example, Seattle METRO's system is set up in a command and control model, while WSDOT's is less hierarchical. In addition, these two units do not have a history of joint development efforts. Nevertheless, a prototype of BusView was constructed with components and cooperation from these two large and complex agencies, despite the fact that they are geographically separate, philosophically distinct, and highly conscious of data integrity.

With future implementation, we anticipate benefits for both the transit operator and the traveling public. On the transit operator side the ability to generalize the geographically limited command and control AVL systems will allow the transit carrier to make real time coach information available to a range of internal users beyond the six consoles constructed for the command and control center. This should allow planners to use the real time coach information in positioning stops and planning routes, as well as allow systems personnel to evaluate the accuracy of the AVL information and install or reposition communications infrastructure to guarantee the desired level of accuracy. These functions are unavailable in the closed command and control environment.

As for the traveling public, a range of possible benefits are possible, including:

1) Increased productivity: If the transit rider can be told when the coach will be at their stop in real time, they are less likely to leave their work site earlier than necessary to allow for the perceived wait time needed to assure catching the coach of their choice.
2) Reduced stress: If the transit user can be assured that the coach she/he is planning to ride has not passed their chosen stop and that it will be arriving in a timely manner, the stress level associated with the lack of information will be reduced
3) Increased public safety: Providing timely information to transit riders will allow them to spend less time in potential dangerous waiting situations.
4) Increased ridership and mode change: The above benefits would lead to a perception that transit is responsive to public needs and that transit is an attractive alternative to SOV travel. This would reduce congestion as well as have a beneficial impact on environmental pollution.

4.2 Additional Issues

An additional issue which complicated the design and development of a BusView prototype was the fact that the AVL system purchased by METRO is a closed proprietary system. Using the eavesdropping technique described above, we were able to obtain coach distance along planned routes as well as some rudimentary status information. However, the proprietary arrangement made it difficult to fine tune the system. For example, when a coach is off route for any reason (e.g. the driver is lost, static rerouting as a result of weather conditions or planned events, or dynamic rerouting in response to congestion or incidents) the present AVL system is unable to place the vehicle geographically. This limitation also applies to non routed para-transit such as that used in response to the ADA legislation. In addition METRO has not as of yet had a real time validation of location on a large scale. Until issues like these can be dealt with, METRO is understandably reluctant to broadcast real time coach location information to the general public for fear of bad publicity or liability from mis-information.

Another issue, yet to be fully addressed, is one of usability. The scope of a test of real time transit location information has been limited to users at the University of Washington who have agreed to participate in an experiment in delivery of APTS information. We are just embarking on obtaining user feedback in a controlled environment at the University of Washington. It is hoped that through iterative design techniques and user feedback, we will produce a prototype that effectively delivers transit information capable of achieving the benefits enumerated.

APPENDIX D: 1999 ITS America Paper

BusView and Transit Watch: Two Products from the Seattle SMART TREK Model Deployment Initiative

D. J. Dailey, G. Fisher, S. Maclean

The Seattle model deployment initiative has created a new technology for data sharing in an ITS environment. This technology, the ITS Backbone that uses Self Describing Data,[1] creates a framework in which to build applications. These applications use Self Describing Data (SDD) to obtain real-time data over the Internet to perform functions ranging from Advanced Traffic Management Systems (ATMS) to Advanced Traveler Information Systems (ATIS). Since the applications depend only on knowledge of SDD, they are portable to any jurisdiction where SDD is employed as the data transfer support.[2] This paper describes the implementation of two such portable applications.

INTRODUCTION

As part of the Seattle Smart Trek model deployment, two new applications where created to provide real-time transit information. The two venues where transit riders need information are (1) on the desktop and (2) at the transit center. The first project, Busview, is a desktop display of the real-time location of all the transit vehicles operated by the regional transit carrier, which operates one of the largest automatic vehicle location system (AVL)

[1] http://www.its.washington.edu/bbone/
[2] SDD information and software are available for download from the ITS Backbone page noted above.

fleets of vehicles in the United States. The second project is Transit Watch, which is a real-time arrival prediction system suitable for deployment in transit centers. Both of these applications are designed to operate over the Internet as Java applets. Both are designed to be sufficiently general so that they can easily be ported to other cities. This paper describes the two applets and the underlying information technology that makes them possible.

BUSVIEW: http://www.its.washington.edu/busview

The first project, Busview Plus, implements a Java applet to display real-time transit vehicle locations on a variety of computing and operating system platforms. Busview Plus is platform-independent with the goal of making transit vehicle location information accessible to anyone on the Internet. An additional goal is to develop an interactive interface that promotes modal change and encourages the use of transit.

The Busview Plus project designed and demonstrated a system that displays real-time transit coach location on a digital map to the Internet community. This project (1) designed an advanced graphical transit information system using Metro’s existing AVL system and the Puget Sound’s regional ITS backbone, (2) created a world wide web page to launch the application, and (3) demonstrated the system’s viability by providing real-time bus location information to individuals on their personal web browsers. This system, a graphical Advanced Public Transportation System (APTS) for the Puget Sound region, is constructed in an open systems model that uses a distributed computing environment. The Busview application, from the University of Washington, leverages a previously installed and expensive Automatic Vehicle Location (AVL) System operated by the transit carrier to create a widely available application to provide real-time transit information.

System Architecture

The Busview applet is built using the component architecture described in [1]. This architecture has been instantiated as the ITS Backbone of the Smart Trek MDI project. In this model, the graphical user interface (GUI) is only one part of the overall application environment. The component model includes the notion of a data stream (SDD) flowing sequentially through a series of components that perform data fusion activities on the data stream. The last of these components (the Sink Component from [1]) is the GUI applet called Busview. Thus the Busview project focused on the component that the users touch, the GUI (often thought of as the application in other environments).

The overall architecture for Busview is shown schematically in Figure 1. It consists of a series of components that operate on the data; each of these components is represented by a rectangle in Figure 1. To explain the overall operation, we describe several of these components.

The transit carrier's automatic vehicle location system is the sensor for the Source component. In this case, the Source component (labeled *AVLUW*) eavesdrops on communication between components of a proprietary AVL system used by the transit carrier's operations staff to (1) measure schedule adherence and (2) provide the transit mangers a location in the event of an incident. The data in this case is distance along a planned path, which, with knowledge of the routes and several coordinate transformations, can be used to estimate location.

AVLUW is an Operator component located on a computer in the AVL operations center of the transit carrier. It limits the information that can leave the agency's operations center. In this case, the numerical identification of the driver is in the data received by *AVLUW,* but it is removed for privacy reasons before the data is allowed out of the physical control of the transit agency. This component is also behind the firewalls protecting the transit AVL system from Internet abuse. Once again, this Operator component provides network security and agency autonomy, as well as removing information from the data stream.

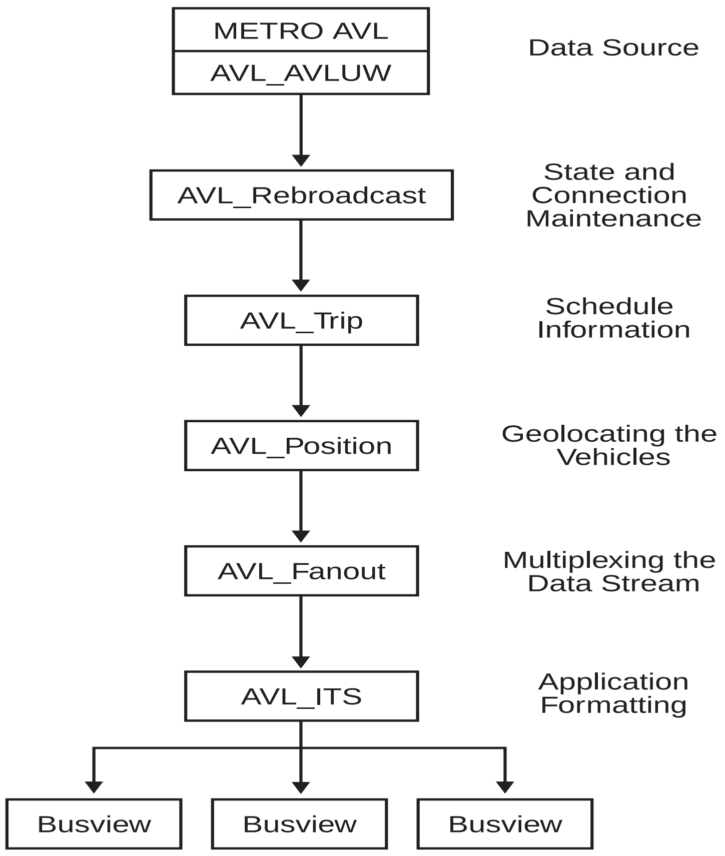


Figure 1: Data Flow and Architecture of Busview

The raw data (a distance along a known path) is multiplexed by a Redistributor component (labeled *AVL_Rebroadcast*) to make it widely available. The data from *AVL_Rebroadcast* is passed to an Operator component (labeled *AVL_Trip*) that uses information in the AVL data packet to associate a specific vehicle with a specific trip in the transit schedule. The data from the AVL system arrives at an average rate of 11-coaches-per-second, and a 20,000-record database of scheduled trips is searched for each coach to obtain the corresponding schedule information. This trip information is added to the record for each coach and made available to the downstream clients. The data from *AVL_Trip* is passed to an Operator (labeled *AVL_Position*) that can use the data (information about planned routes, digital maps, and coordinate transformations) to calculate a latitude and longitude value for the transit (probe) vehicles. The calculation of position requires (1) identifying the particular coordinate lists in one of 2500+ files that represent every possible vehicle route, (2) selecting the segment within the list on which the vehicle is estimated to exist, and (3) using a Newton's method to perform an inverse Lambert projection from the proprietary coordinate system of the AVL system.

Downstream of the *AVL_Position* server is a Redistributor labeled AVL*_Fanout* that multiplexes the data stream containing the vehicle positions to a variety of users. This Redistributor frees the upstream process from multiplexing in addition to performing the positioning solution. The Server component (labeled *AVL_Its*) creates a customized data stream for the Busview presentation. The connection between this server and the Busview sink can potentially be over slower media. This component minimizes the amount of data that need be sent per update of each transit vehicle on the Busview screen. A digital map and

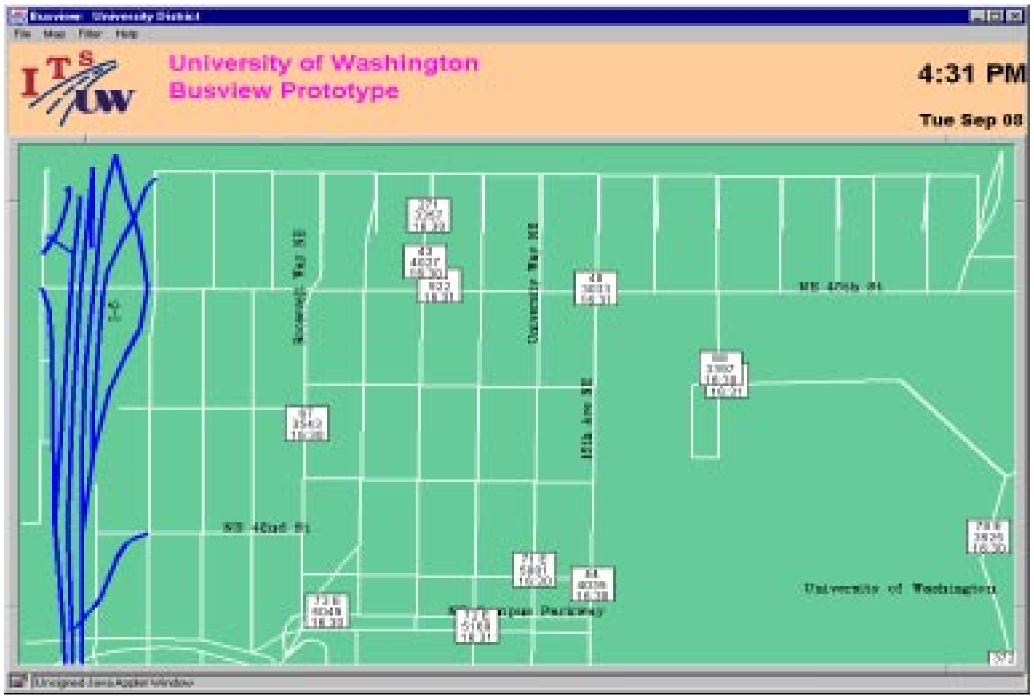
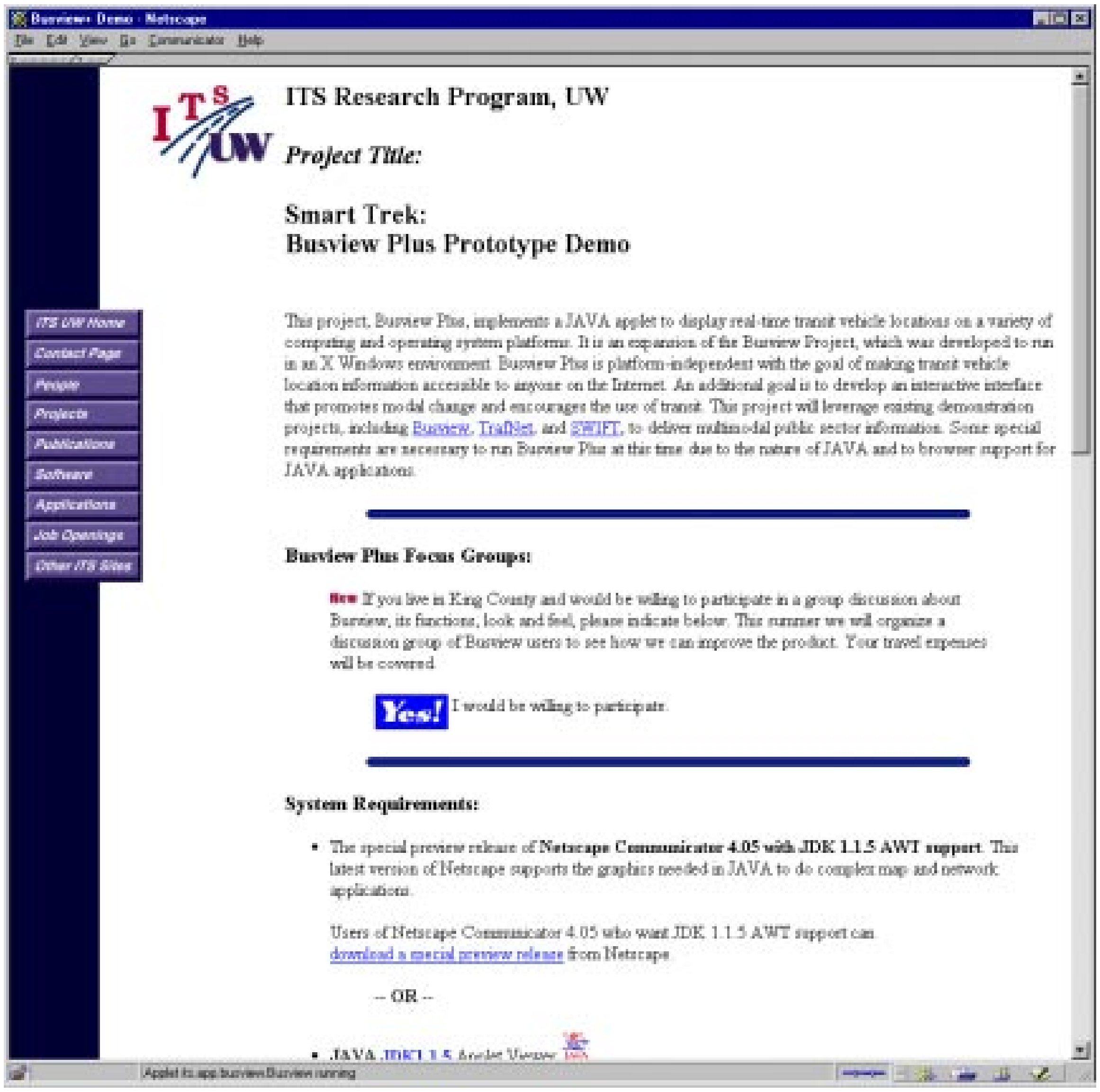


Figure 2: Busview screen

the vehicle location information, along with schedule information, are displayed graphically on the Busview screen, creating an information system for transit riders.

The Busview graphical user interface (GUI) shown in Figure 2 operates as a Java applet. This GUI produces a representation of map data on a browser that may be located anywhere on the Internet. A web page to facilitate the use of the Busview applet (see Figure 3) is available at http://www.its.washington.edu/busview.



ITS Research Program, UW

Project Title:

Smart Trek:
Busview Plus Prototype Demo

This project, Busview Plus, implements a JAVA applet to display real-time transit vehicle locations on a variety of computing and operating system platforms. It is an expansion of the Busview Project, which was developed to run in an X Windows environment. Busview Plus is platform-independent with the goal of making transit vehicle location information accessible to anyone on the Internet. An additional goal is to develop an interactive interface that promotes modal change and encourages the use of transit. This project will leverage existing demonstration projects, including Busview, TrafNet, and SWIFT, to deliver multimodal public sector information. Some special requirements are necessary to run Busview Plus at this time due to the nature of JAVA and to browser support for JAVA applications.

Busview Plus Focus Groups:

New If you live in King County and would be willing to participate in a group discussion about Busview, its functions, look and feel, please indicate below. This summer we will organize a discussion group of Busview users to see how we can improve the product. Your travel expenses will be covered.

I would be willing to participate.

System Requirements:

- The special preview release of **Netscape Communicator 4.05 with JDK 1.1.5 AWT support**. This latest version of Netscape supports the graphics needed in JAVA to do complex map and network applications.

  Users of Netscape Communicator 4.05 who want JDK 1.1.5 AWT support can download a special preview release from Netscape.

  -- OR --

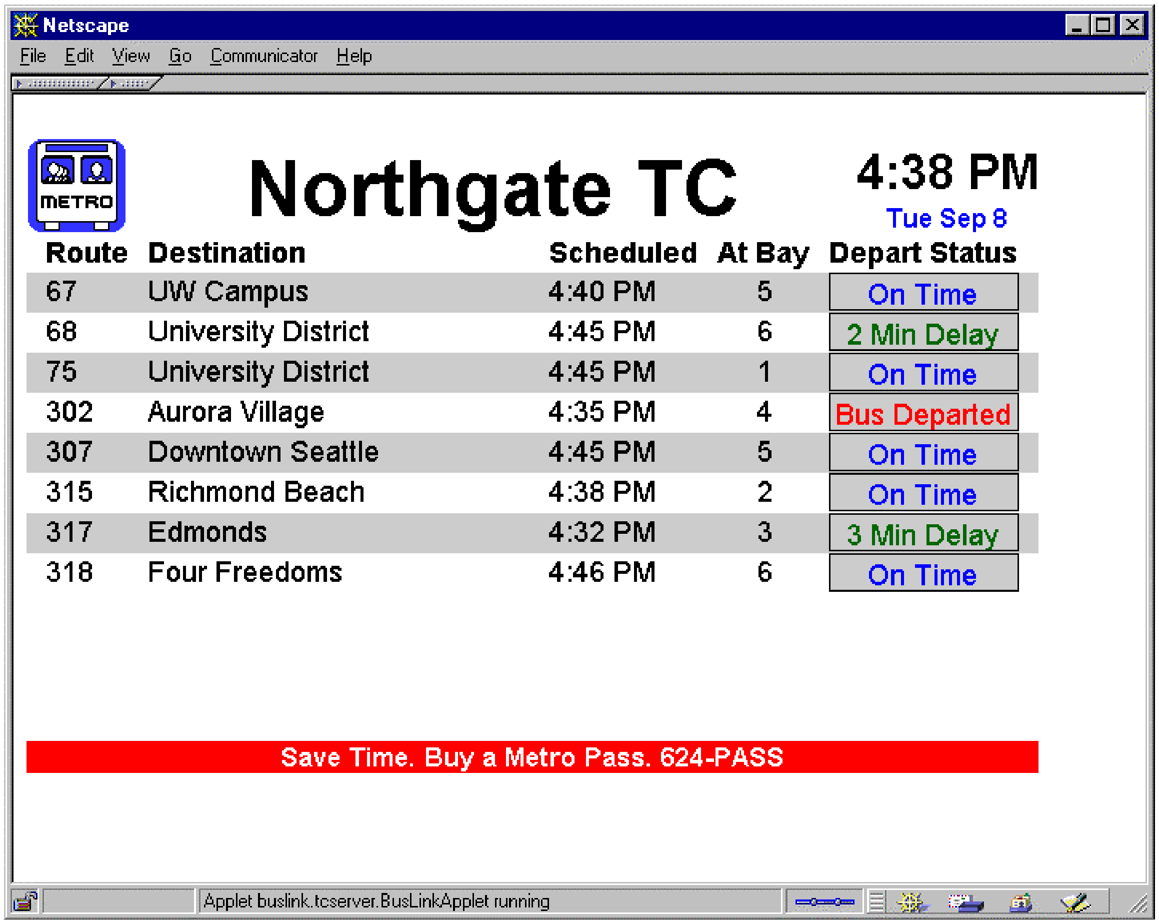
- JAVA JDK1.1.5 Applet Viewer

Figure 3: Busview WWW launch page

The second project presented here is an applet to provide real-time information at the transit station.

Transit Watch

This project deployed an Acvanced Public Transportation System/Advanced Traveler Information System (APTS/ATIS) that provides a prediction of the arrival status of schedule transit coaches. This prediction is quantized into four states: (1) ONTIME, (2) Delayed N minutes, (3) Departed, and (4) No Information, as shown in Figure 4. The transit carrier’s customer relations specialist selected the delay categorizations. The goal of the project is to develop an interface that promotes the use of transit by reducing the stress inherent in



Northgate TC

4:38 PM
Tue Sep 8

Route	Destination	Scheduled	At Bay	Depart Status
67	UW Campus	4:40 PM	5	On Time
68	University District	4:45 PM	6	2 Min Delay
75	University District	4:45 PM	1	On Time
302	Aurora Village	4:35 PM	4	Bus Departed
307	Downtown Seattle	4:45 PM	5	On Time
315	Richmond Beach	4:38 PM	2	On Time
317	Edmonds	4:32 PM	3	3 Min Delay
318	Four Freedoms	4:46 PM	6	On Time

Save Time. Buy a Metro Pass. 624-PASS

Applet buslink.tcserver.BusLinkApplet running

Figure 4: Transit Watch screen

transfers. This project leverages the ITS Backbone and Self Describing Data components of the SmartTrek MDI project. This project was originally designed to be deployed at three transit centers, but it has since been made available on the Internet at http://www.its.washington.edu/TransitWatch.

System Architecture

Transit Watch is a multi-component client/server database application to disseminate schedule and performance information to Metro riders at transit centers. The Transit Watch application is comprised of four main elements: (1) a database engine, (2) a Predictor Module (labeled "Predictor" in Figure 5), (3) a Display Server, and (4) client applets.

The database engine is central to the application (as shown in Figure 5). The data storage, relational queries, and serialization capabilities are used to synchronize the Predictor

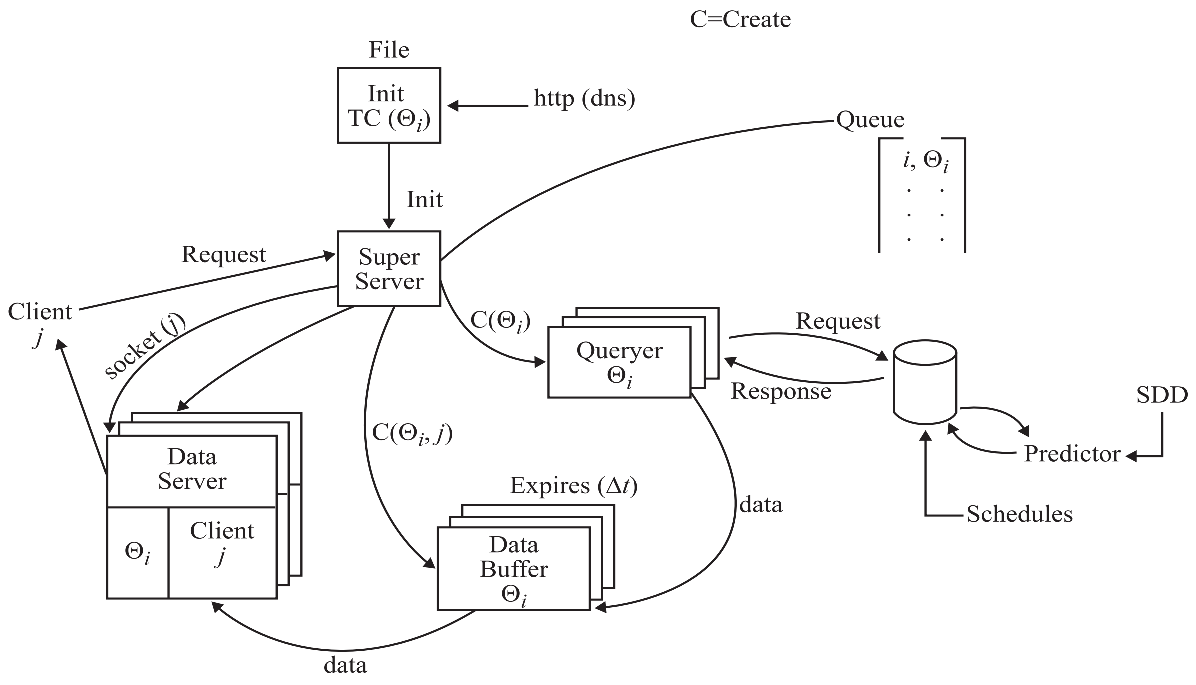


Figure 5: Transit Watch design

and Display Server components. The database stores information from the AVL stream necessary for the Predictor to make arrival time predictions as well as storing the resulting predictions for use by the Display Server. In this way, the Predictor and Display servers operate asynchronously and autonomously but share a common set of validated data.

The Predictor is shown on the right side of Figure 5. To make predictions, several pieces of information are necessary. The required data includes: (1) the real-time locations of the vehicle from the AVL system, (2) the scheduled travel plan for the vehicle, (3) the geographical route over which the vehicle travels, (4) mean travel speeds for vehicles, and (5) past travel histories for the route of interest. These are combined in a prediction algorithm to estimate the time until arrival at a selected transit center for a particular vehicle on the specific route. The time horizon for this prediction is set such that vehicles scheduled to arrive 30 minutes in the future and those that have departed within five minutes are included. For the Seattle system, this means that approximately 50-100 vehicles are tracked at any one time for a medium-size transit center.

The Predictor obtains the required data from a number of sources. Items (2) and (3) are relatively static based on the transit carrier route planning and are provided by the carrier. Item (1), the real-time data, utilizes an underlying SDD receiver from the SDD2.0.0b6 MDI backbone code. Items (4) and (5) are derived information created as the SDD AVL data is received and processed into the database by the Predictor. These data sources are the basis for creating a reliable prediction algorithm.

The prediction algorithm operates in two modes. The first is to use an a-priori travel speed, with the time and location of the transit vehicle, to make a linear prediction of the arrival time. This linear prediction is returned to the database to make it accessible to the Display Server. This simple form of prediction is the basis for most systems of this type.

The Predictor also operates in a second “statistical mode.” As each vehicle is observed approaching the transit center of interest, the data points (position and time of observation) are placed into the database. When the vehicle arrives at the transit center, the database is updated to include the tuple (position, time of observation, delay between observation and arrival). As the system operates, it collects data for each of the vehicles on each of the scheduled routes. The Predictor examines the database each time a vehicle of interest is observed in the AVL data stream. If sufficient observations of past trips have been made in a small geographical region near this newest point, the Predictor can return a mean estimate of arrival time from these data points. In addition, an estimate of the variance for vehicles at this distance from the transit center can be produced. The statistics of the arrival time data are approximated as normal in the treatment of the errors in the data. This approximation is justified based on the results of a Kolmonogrov-Smirnoff distribution membership test applied to the arrival time data. If operating in statistical mode, the results of this prediction process are returned to the database.

Once reliable predictions are available in the database, the second specialized component of the Transit Watch architecture is used to distribute the information either to applet clients at transit centers or on the Internet. The Display Server is shown on the right of Figure 5. This component is made up of several elements: (1) a super server to invoke the other elements as needed, (2) a Data Server for each client display (denoted by *j*), (3) a Data Buffer for each transit station (denoted by *i*), and (4) a Query Client for each transit center. The clients (a display computer at a transit center) make a request to the Super Server for a display of the information for one particular transit center. The Super Server then (1) creates a Query Client to access the database for the data for that transit center, (2) creates a Data Buffer to asychronously hold the information for the transit center, and (3) creates a Data Server for the transit center of interest for the client requesting the data. This structure is designed to support a large number of clients in real time. The client screen is shown in Figure 4.

SUMMARY

SmartTrek, the Seattle MDI, deployed a new technology, Self Describing Data, and created two example applications to use this technology. The applications, Busview and Transit Watch, are directed at transit users in Seattle but are examples of applications that can be generalized to any region, thus providing a model for ITS application deployment.

REFERENCES

1. Dailey, D. J., M. P. Haselkorn, and D. Meyers. "A Structured Approach to Developing Real-Time, Distributed Network Applications for ITS Deployment." *ITS Journal*, Vol. 3, No. 3, pp. 163-80, 1996.