Final Research Report
Research Project T9903 Task 78
Timesaver Backbone

Smart Trek: A Model Deployment Initiative

by

D.J. Dailey
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 N.E. 45th Street, Suite 535
Seattle, Washington 98105-4631

Washington State Department of Transportation
Technical Monitor
Pete Briglia

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

May 2001

TECHNICAL REPORT STANDARD TITLE PAGE

1. REPORT NO.	2. GOVERNMENT ACCESSION NO.	3. RECIPIENT'S CATALOG NO.
WA-RD 505.1		

4. TITLE AND SUBTITLE	5. REPORT DATE
Smart Trek: A Model Deployment Initiative	May 2001
	6. PERFORMING ORGANIZATION CODE

7. AUTHOR(S)	8. PERFORMING ORGANIZATION REPORT NO.
D.J. Dailey	

9. PERFORMING ORGANIZATION NAME AND ADDRESS	10. WORK UNIT NO.
Washington State Transportation Center (TRAC) University of Washington, Bx 354802 University District Building; 1107 NE 45th Street, Suite 535 Seattle, Washington 98105-4631	11. CONTRACT OR GRANT NO. Agreement T9903, Task 78

12. SPONSORING AGENCY NAME AND ADDRESS	13. TYPE OF REPORT AND PERIOD COVERED
Washington State Department of Transportation Transportation Building, MS 7370 Olympia, Washington 98504-7370 Project Manager: Gary Ray, 360-705-7975	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 SmartTrek project was implemented by a consortium of government agencies, private sector firms, and the University of Washington. The project resulted in a variety of intelligent transportation system (ITS) applications that help traffic management personnel and the traveling public. This report documents the components of the SmartTrek project implemented by the University of Washington. It provides a summary and evaluation of those applications and is a "one-stop shopping" overview of the UW SmartTrek activities.

17. KEY WORDS	18. DISTRIBUTION STATEMENT
Intelligent transportation system, advanced traffic management system, advanced public transportation system, advanced traveler information system	No restrictions. This document is available to the public through the National Technical Information Service, Springfield, VA 22616

19. SECURITY CLASSIF. (of this report)	20. SECURITY CLASSIF. (of this page)	21. NO. OF PAGES	22. PRICE
None	None		

DISCLAIMER

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the data presented herein. The contents do not necessarily reflect the official views or policies of the Washington State Transportation Commission, Department of Transportation, or the Federal Highway Administration. This report does not constitute a standard, specification, or regulation.

Table of Contents

Traffic Channel

Backbone

MyBus

List of Tables

List of Figures

Glossary

Applet A small computer application that has limited features, requires limited memory resources, and is usually portable between operating systems and across the Internet.

API Applications Programming Interface–The interface (calling conventions) by which an application program accesses an operating system and other services.

APTS Advanced Public Transportation System–Groups and systems of technologies that support the use of public transportation systems and shared ride transportation modes.

ATIS advanced traveler information system–Groups and systems of technologies that aid in the collection, collation, and dissemination of traveler information before and during trips.

ATMS Advanced Traffic Management System–An array of institutional, human, hardware, and software components designed to monitor, control, and manage traffic on streets and highways.

AVL Automatic Vehicle Location–The use of electronic tags and readers to locate the position of vehicles on the road in real time. Often used for commercial fleets or public transportation.

Backbone, ITS A set of protocols designed to tie ITS applications together. These applications extract ITS data from various agencies, modify those data, and redistribute them. This term can also include the physical connections that tie distributed ITS systems together.

BER Basic Encoding Rules–ASN.1 encoding rules for producing self-identifying and self-delimiting transfer syntax for data structures described in ASN.1 notations. BER is a self-identifying and self-delimiting encoding scheme, which means that each data value can be identified, extracted, and decoded individually.

CCTV Closed-Circuit Television–Cameras that give traffic management personnel real-time views of traffic conditions around the region.

Data dictionary A data structure that stores meta-data, i.e,. data about data. Most generally, it is a set of data descriptions that can be shared by several applications. Usually it means a table in a database that stores the names, field types, length, and other characteristics of the fields in the database tables. An active data dictionary is automatically updated as changes occur in the database. A passive data dictionary must be manually updated.

DBMS Database Management System–A software system that facilitates the creation, maintenance, and use of an electronic database

Fusion, data The process of combining traffic information collected from numerous sources, analyzing the data to check for errors, and formatting the data into a standard format for users.

GIS Geographic Information System–A computer system for capturing, storing, checking, integrating, manipulating, analyzing, and displaying data related to positions on the Earth's surface. Typically, a GIS is used for handling maps.

GUI Graphical User Interface–The use of pictures rather than just words to represent the input and output of a program. A program with a GUI runs under some windowing system.

HTML Hypertext Markup Language–A markup language used to structure text and multimedia documents and to set up hyperlinks between documents, used extensively on the World Wide Web.

I2B ITS information backbone–(see backbone, ITS)

ISDN Integrated Services Digital Network–A set of communications standards allowing a single wire or optical fiber to carry voice, digital network services, and video. ISDN is intended to eventually replace the plain old telephone system.

ISP Information Service Provider–A company or agency that obtains data, adds value to the data through customization or packaging, and then provides those data to customers.

ITS Intelligent Transportation Systems–ITS involves integrated applications of advanced surveillance, communications, computer, display, and control process technologies on the roadway network, in the vehicle, and for transit.

JDBC Java Database Connectivity–Part of the Java Development Kit that defines an application programming interface for Java for standard SQL access to databases from Java programs.

Kalman filter A set of mathematical equations that provides an efficient computational (recursive) solution of the least-squares method. The filter is very powerful in several aspects: it supports estimations of past, present, and even future states, and it can do so even when the precise nature of the modeled system is unknown.

KC Metro King County Metro Transit–Metro is a division of the King County Department of Transportation.

Kolmonogrov-Smirnoff test Standard distribution membership test, used for continuous data as a function of a single variable. This test is applicable to unbinned (continuous) distributions that are funtions of a single independent variable, that is, to data sets where each data point can be associated with a single number (lifetime of each lightbulb when it burns out, or declination of each star).

Lambert projection A conical map projection used to map geodetic points onto a flat map. The spherical (globe) grid is projected onto a flat plane, thus it is also called a plane projection. It is also referred to as an Azimuthal projection. The poles are the “normal aspect” (the viewpoint or perspective) which results in the simplest projected grid for this family of projections. That is, the plane is normally placed above the north or south pole. Normally only one hemisphere, or a portion of it, is represented on azimuthal projections. When projected from the centre of the globe with the normal aspect, the typical grid appearance for azimuthal projections shows parallels forming concentric circles, while meridians radiate out from the centre.

Meta-data Data about data. In data processing, meta-data are definitional data that provide information about or documentation of other data managed within an application or environment. For example, meta-data would document data about data elements or attributes (name, size, data type, etc.), records or data structures (length, fields, columns, etc.), and data (where they are located, how they are associated, ownership).

	They may include descriptive information about the context, quality and condition, or characteristics of the data.
MDI	Model Deployment Initiative (Seattle Smart Trek)–Used interchangeably in this report with MMDI.
MMDI	Metropolitan Model Deployment Initiative (Seattle Smart Trek)–A federally funded transportation program under which four metropolitan areas-Phoenix, Seattle, San Antonio, and The New York city metropolitan area-were chosen to showcase the deployment of ATIS technologies
msec	millisecond
Newton's method	A technique for finding the roots of an equation, that is, the values where the equation equals zero. You begin by choosing a point in the complex plane as an initial guess, and then iterating to get better and better guesses.
NTCIP	National Transportation Communications for ITS Protocol–The NTCIP is a family of standards that provides both the rules for communicating (called protocols) and the vocabulary (called objects) necessary to allow electronic traffic control equipment from different manufacturers to operate with each other as a system. The NTCIP is the first set of standards for the transportation industry that allows traffic control systems to be built using a "mix and match" approach with equipment from different manufacturers. Therefore, NTCIP standards reduce the need for reliance on specific equipment vendors and customized one-of-a-kind software. To assure both manufacturer and user community support, NTCIP is a joint product of the National Electronics Manufacturers Association (NEMA), the American Association of State Highway and Transportation Officials (AASHTO), and the Institute of Transportation Engineers (ITE).
NTSC	National Television Standards Committee–The body defining the television video signal format used in the US.
PDA	Personal Data Assistant
PNG	Portable Network Graphics–An extensible file format for the lossless, portable, well-compressed storage of raster images. PNG provides a patent-free replacement for GIF and can also replace many common uses of TIFF. PNG is designed for on-line viewing applications, such as the World Wide Web.
PSRC	Puget Sound Regional Council–The PSRC is an association of cities, towns, counties, ports, and state agencies that serves as a forum for developing policies and making decisions about regional growth management, economic and transportation issues in the four-county central Puget Sound region.
Quantize	To limit the possible values of (a magnitude or quantity) to a discrete set of values by quantum mechanical rules.
RGB	red, green, blue–Often used as a synonym for color, as in "RGB monitor," as opposed to monochrome (black and white).
SDD	Self-Describing Data–A process of electronically sending a data stream containing not only specially formatted raw data but also a data dictionary of information about where the data come from and what they mean (i.e.,meta-data).
SDTS	Spatial Data Transfer Standard–A practical and effective vehicle for the exchange of spatial data between different computing platforms. It is designed specifically as a

	format for the transfer of spatial data–not for the direct use of the data. The full SDTS specification creates a framework for spatial data transfer by defining different "levels," from the real world to the physical encoding of the data. SDTS was ratified by the National Institute of Standards and Technology (NIST) as a Federal Information Processing Standard (FIPS 173) in 1992. Compliance with FIPS 173 by federal agencies became mandatory in 1994.
SNMP	Simple Network Management Protocol–The Internet standard protocol, defined in STD 15, RFC 1157, developed to manage nodes on an IP network. SNMP is not limited to TCP/IP. It can be used to manage and monitor all sorts of equipment, including computers, routers, wiring hubs, toasters and jukeboxes.
SNTP	Simple Network Time Protocol–SNTP is used to synchronize computer clocks on local area networks.
SQL	Structured Query Language–An industry-standard language for creating, updating, and querying relational database management systems.
TCIP	Transit Communication Interface Profiles–Computer interface standards for use in the US transit industry.
TDAD	Traffic Data Acquisition and Distribution–Research project led by Daniel J. Dailey that created a data mine of traffic data.
TMS	Traffic Management System–System of various components, such as ramp meters, variable message signs, highway advisory radio, telephone hotline numbers, and other methods, designed to monitor and manage highway traffic.
TNP	Transportation Network Profile–The Transportation Network Profile (TNP) contains specifications for an SDTS profile for use with geographic vector data with network topology.
TRAC	Washington State Transportation Center–TRAC provides a link among the government, university researchers, and the private sector. Much of their research is funded by the Washington State Department of Transportation (WSDOT), and TRAC acts as a liaison, connecting those who need applied research at WSDOT with those best suited to conduct it at the universities.
TSMC	Traffic Systems Management Center–Center that houses human, hardware, and software components designed to monitor, control, and manage traffic on the highways.
Tuple	Toyohashi University Parallel Lisp Environment–A parallel Lisp (a programming language designed to process data consisting of lists) based on Kyoto Common Lisp (an implementation of Common Lisp written in C to run under Unix-like operating systems).
UWTV	University of Washington cable television channel
VGA	Video Graphics Array–A display standard for IBM PCs, with 640 x 480 pixels in 16 colors and a 4:3 aspect ratio. There is also a text mode with 720 x 400 pixels.
WSDOT	Washington State Department of Transportation

Introduction

The SmartTrek project was implemented by a consortium of government agencies, private sector firms and the University of Washington. The project resulted in a variety of Intelligent Transportation Systems (ITS) applications that help traffic management personnel and the traveling public. This report provides documentation on the components of the SmartTrek project implemented by the University of Washington. It provides a summary of the applications from the UW and provides evaluation of those applications. This report is a "one stop shopping" overview of the UW SmartTrek activities.

University research activities often consist of two components, (1) the pure research that creates new knowledge and is documented by publications and (2) implementation of the research often demonstrated by implementing the new ideas in software. This report combines published papers and software descriptions to provide an overall picture of the activities completed at the University of Washington. In addition it includes work by other SmartTrek partners that undertook evaluation of the functionality and effectiveness of the implementations built at the UW.

The format of the report is a series of chapters made up of these papers and software descriptions.

We first provide a paper presented at the 6th World Congress on ITS that describes the Busview and Transit Watch implementation that focuses principally on Transit Watch. An evaluation of the Transit Watch project, published as the "Metropolitan Deployment Initiative Evaluation" by the US DOT Joint Program office, is included next. It documents the effectiveness of providing real-time information at transit centers.

Next, a chapter that describes the aspects of Busview (www.busview.org) that were influenced by SmartTrek is included. This information is excerpted from a larger report, WA-RD 467.1, that includes the entire development history of Busview. During the course of the SmartTrek project, large volumes of email were received regarding the Busview applet, and it was all very complimentary. Some typical comments are included in this report. A focus group was convened by Pacific Rim Resources to evaluate Busview, and the outcomes of that group are included as the next chapter. Finally, a chapter reports the detailed usage statistics for Busview from September 1, 1998, through January 2001.

The next chapter describes the Traffic Channel deployed on UWTV2, channel 76, on the AT&T cable network in Seattle. This project created a mixed media system to provide a professional quality information channel that does not require any ongoing staff effort. The application design and the selection of presentation materials are detailed in the material extracted from the Traffic Channel web site, www.its.washington.edu/trafchan. The following chapter provides an evaluation of the Traffic Channel and is excerpted from the MMDI evaluation report from the Joint Program Office of USDOT.

The ITS Backbone, over which the real-time data flow, was a central component of the SmartTrek activities. The next chapter, extracted from a TRB presentation and a Transportation Research Record paper, details the backbone concepts and implementation. This is followed by a chapter describing a case study undertaken by Virginia Tech to evaluate the ITS Backbone.

An additional application, MyBus (www.mybus.org) was developed during the SmartTrek project. It is designed to provide a real-time prediction of transit vehicle arrival for every vehicle (~1200) over a large geographical area (1500 sq.mi.) for many stops (~2000). This project created an optimal prediction algorithm that is documented by the TRB 2001 paper in this report. The actual design and construction of the application, as extracted from a 2000 World Congress paper, is also included in this report. Another paper presented at the 2000 ITS World Congress demonstrates the usage of Transit Communication Interface Profile (TCIP) to construct a database from which all of the transit applications are built. The evaluation of MyBus's effectiveness takes the form of usage statistics. The MyBus website was viewed over 1.7 million times in January 2001.

Overall, this report provides a detailed summary of the UW contribution to the SmartTrek MMDI. The UW SmartTrek project component created both a set of ITS applications and a set of publications that document a variety of new technologies to help people travel smarter.

Transit Watch

1. BusView and Transit Watch: an Update on Two Products from the Seattle SMART TREK Model Deployment Initiative [1]

1.1 Introduction

As part of the Seattle Smart Trek model deployment, two new applications were 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) 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.

1.2 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 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 project designed and demonstrated a system that displays real-time transit coach location to the Internet community on both a digital map and a route progress bar. This project (1) designed an advanced graphical transit information system using Metro's existing AVL system (Metro Transit is the regional transit carrier for King County, Washington) and the Puget Sound's regional ITS backbone (http://www.its.washington.edu/bbone/), (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.

1.2.1 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 1-1. It consists of a series of components that operate on the data; each of these components is represented by a rectangle in Figure 1-1. To explain the overall operation, we describe several of these components.

---

[1] Paper presented at the 6th World Congress on Intelligent Transport Systems in Toronto, Canada, November 8-12, 1999. Authors: D.J. Dailey, G. Fisher, and S. Maclean.

The transit carrier's automatic vehicle location system is the sensor for the Source component. In this case, the Source component (labeled *METRO AVL*) 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 managers 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.

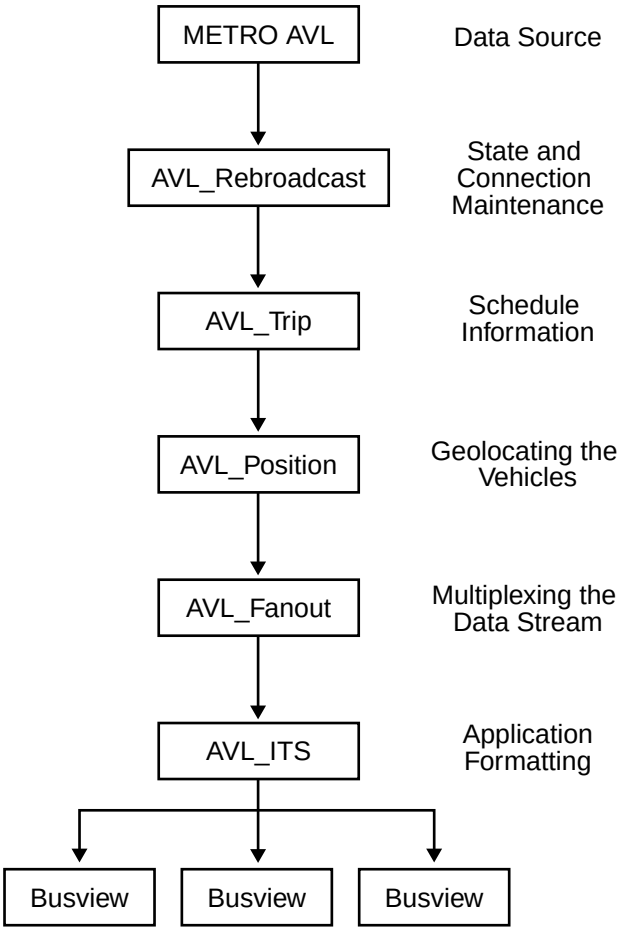


FIgure 1-1. Data flow and architecture of Busview

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 (see the Operator Component definition in (1)) provides network security and agency autonomy, as well as removing information from the data stream.

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 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.

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

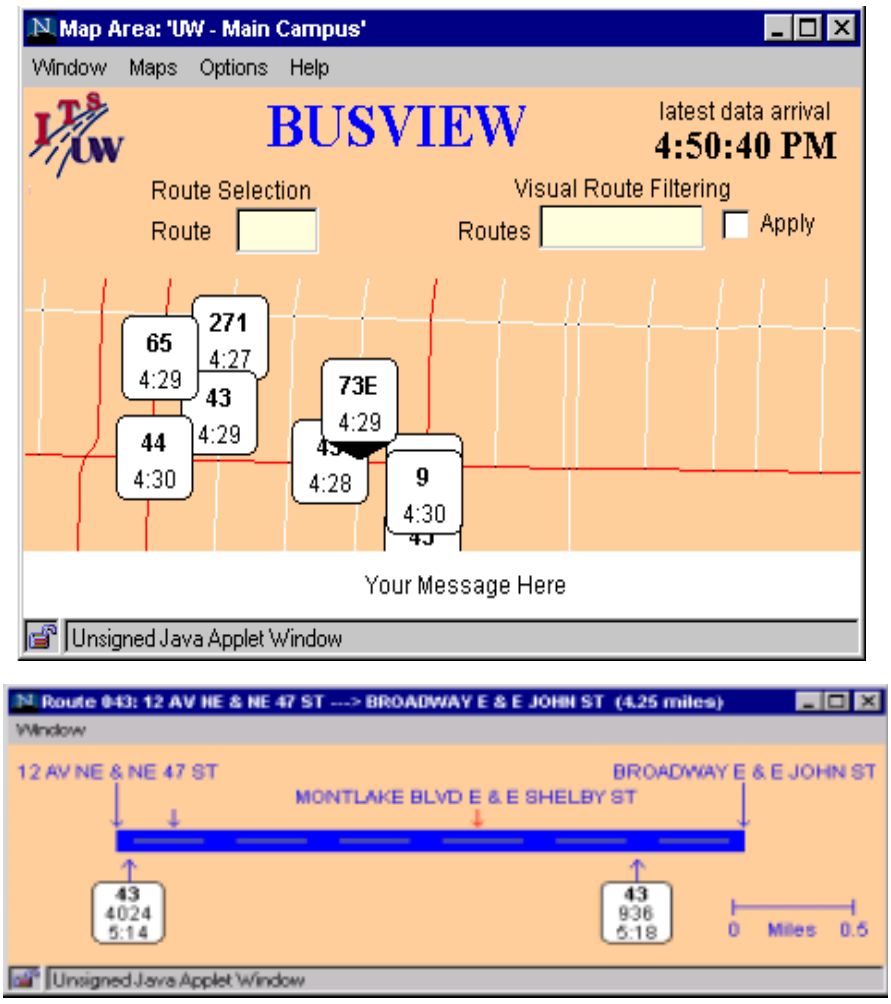


Figure 1-2. Busview screen (top) and progress bar (bottom)

The Busview graphical user interface (GUI) shown at the top of Figure 1-2 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 is available at http://www.its.washington.edu/busviewplus/. In addition, the applet can generate a progress bar description of the transit vehicle's progress along the route. This progress

bar allows for the insertion of a "push-pin" along the vehicle path that sends an alert to the user as the transit vehicle passes the marked point on the route. The bottom graphic in Figure 1-2 shows an example of the Busview progress bar.

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

1.3 Transit Watch - http://www.its.washington.edu/transitwatch

This project deployed an APTS/ATIS that provides a prediction of the arrival status of scheduled 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 1-3. The quantization into these four major catogories was selected by the transit carrier's customer relations specialist. The overall goal of this product is an interface that promotes the use of transit by reducing the stress inherent in 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.

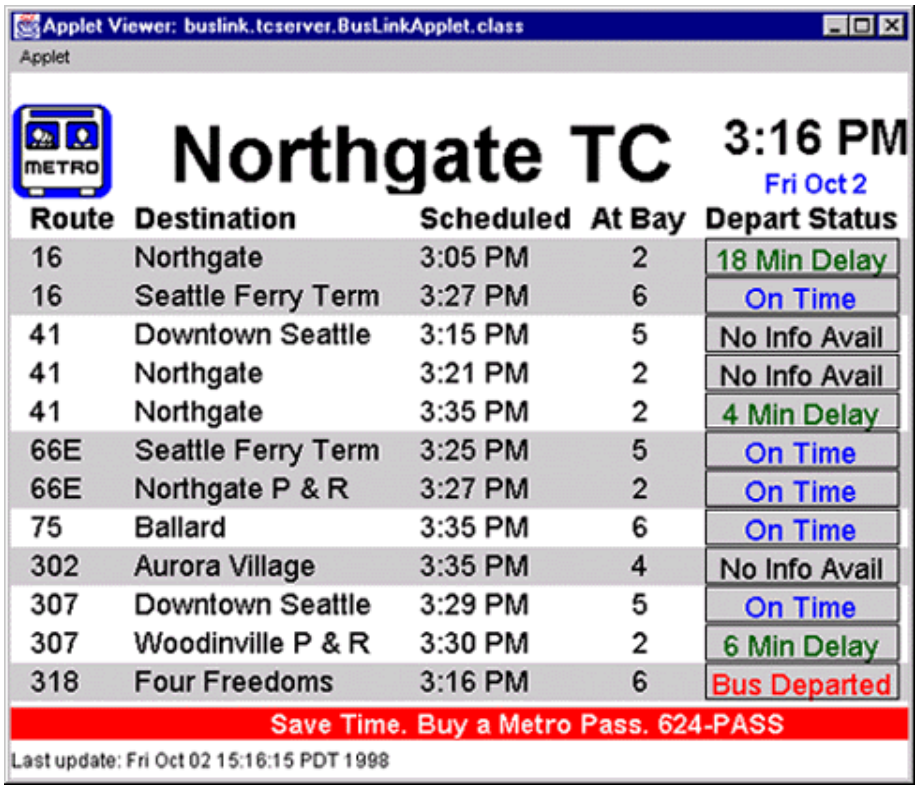


Figure 1-3. Transit Watch screen

1.3.1 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 1-4), (3) a Display Server, and (4) client applets.

The database engine is central to the application (as shown in Figure 1-4). The data storage, relational queries, and serialization capabilities are used to synchronize the Predictor 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.

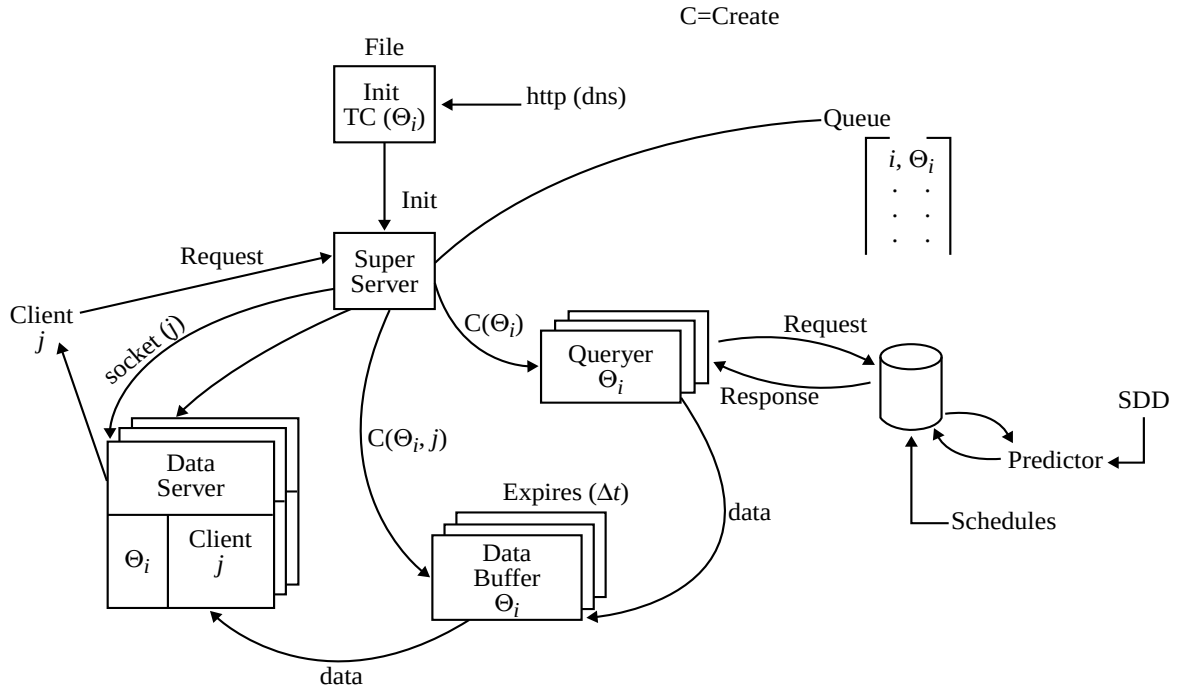


Figure 1-4. Transit Watch design

The Predictor is shown on the right side of Figure 1-4. To make predictions, several pieces of information are necessary. The required data include: (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 left of Figure 1-4. 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 1-3.

1.4 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.

1.5 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.

2. King County "Transit Watch" Evaluation [2]

2.1 Project Description

Transit Watch is a video monitor providing transit information at two transit centers in the Seattle metropolitan region: Northgate and Bellevue. The video monitors were installed in mid-1998 as part of the Smart Trek MMDI program. The monitors provide information about the bay number at which the bus will arrive, the scheduled times of arrival and departure, and the expected *actual* departure times for all of the bus routes using the transfer centers. Actual times are based on information obtained from an Automated Vehicle Location (AVL) system. Transit Watch is shown in Figure 2-1.

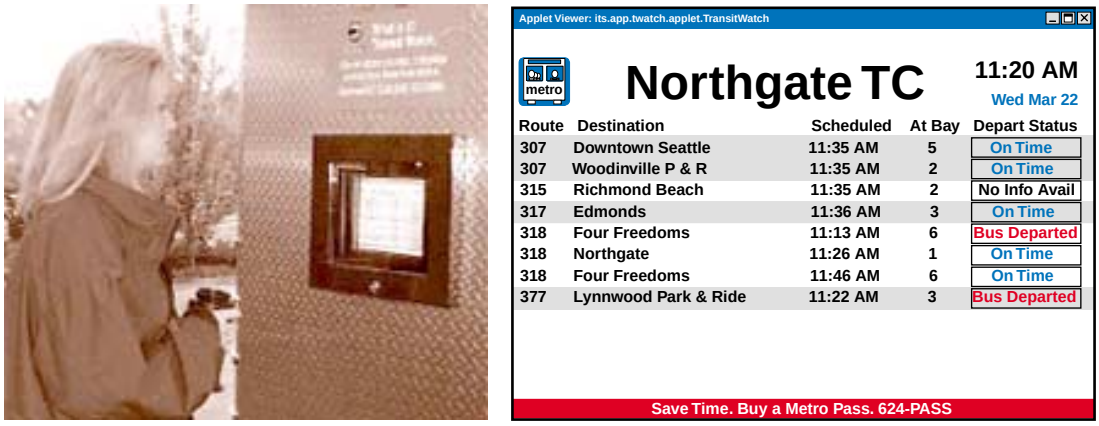


Figure 2-1. A deployed Transit Watch system in Seattle

Figure 2-2. Transit Watch is deployed at the state-of-the-art Northgate Transit Center

[2] Excerpted from Metropolitan Model Deployment Initiative Seattle Evaluation Report Final Draft, Publication No.: FHWA-OP-00-020, May 30, 2000, U.S. Department of Transportation, ITS Joint Program Office, HVH-1, 400 7th Street SW, Washington, DC 20590, Phone: (202) 366-0722, Facsimile: (202) 493-2027

The MMDI costs to deploy the Transit Watch are presented in Table 2-1. The major cost drivers here were the UW development labor costs and the system hardware costs. Annual operations and maintenance costs for the project are estimated to be approximately 19 percent of the deployment costs.

Table 2-1. Transit Watch Cost Estimate [3]

Equipment Description	Non-Recurring	Recurring Costs
King Co. Hardware (2 PC's, 5 monitors, modems, routers)	$ 39,500	
King Co. Facilities Improvements	$ 6,400	
King Co. Boeing Transit Watch	$ 20,000	
King Co. Engineering & Management Support	$ 3,300	
King Co. Software Development Support	$ 5,000	
King Co. Design Consultant	$ 43,000	
King Co. Other Labor	$ 28,000	
UW Total Equipment Purchases (Proposed Budget)	$ 25,000	
UW Total Supplies & Materials (Proposed Budget)	$ 19,900	
UW Development Labor	$ 447,475	
King Co. Maintenance		$ 4,100
Operations Labor (1 UW FTE)		$ 131,250
14% Share of 3 Pentium Workstations & Associated Equipment	$ 3,393	
14% Share of Labor (including indirect costs & benefits)	$ 78,324	
14% Share of Other Direct Costs	$ 3,585	
14% Share of Hardware & Supplies (replaced every 2 years)		$ 1,696
14% Share of Fiber Link & Other Contractual Services		$ 2,428
14% Share of Operations Labor (3 UW FTEs)		$ 40,178
Total	**$ 722,877**	**$ 179,652**

2.2 Evaluation Description

The survey documented in this report was carried out as part of the Customer Satisfaction evaluation of MMDI programs. This survey had four primary objectives:

- To characterize the passengers who use the transit center and Transit Watch.
- To analyze the use of the monitors and to explore possible improvements to the monitor-based system and provide guidance in that planning.
- To assess the satisfaction, value, and behavioral responses of transit riders based on use of Transit Watch and access to real-time bus status information.
- To assess the potential influence of Transit Watch on ridership.

Respondents for the Transit Watch survey were initially recruited in January 1999 at both the Bellevue Transit Center and the Northgate Transit Center. Telephone interviews were conducted between January and March 1999, resulting in a total of 505 completed questionnaires.

[3] Note that a 14% share of the ITS Information Backbone System cost elements have been included here as part of this cost estimate since relevant ITS information is either currently exchanged between this system and the Backbone, or because this system has the potential to interface with the Backbone in the near future.

2.3 Results

2.3.1 Transit Watch Usage, Transit Center User Characteristics and Suggested Improvements

The respondents were representative of people using these two Transit Centers, but we cannot generalize from the experiences of riders at these centers to all riders or locations in the King County Metro system. Three out of every four respondents were aware of Transit Watch. About 22 percent of the entire sample stated that they *always* use Transit Watch. About another one in four (28 percent) said that they use it sometimes, 26 percent said that they rarely use Transit Watch, and 25 percent said they had never seen the monitors.[4]

The regular users — those who said they always use Transit Watch —are slightly younger, slightly more educated (though not statistically significant), and slightly more technologically savvy than the non-users. There is no significant difference in the incomes of Transit Watch users and non-users. Regular and occasional Transit Watch users are likely to use the bus system somewhat more extensively and for a greater variety of purposes than the non-users.

A majority of the respondents were long-term users of the bus service, and most of them (Transit Watch users and non-users alike) indicated a high awareness of bus schedules. Providing *actual bus departure times* is the Transit Watch feature found most useful by the users. The scheduled departure times and route number descriptions also seem to be useful.

2.3.2 Suggestions for Improvement

Although almost half of the aware respondents (as well as the regular and occasional user segments) had no suggestions for improvements to the Transit Watch monitors, those who did, mentioned *improved accuracy* most often.

All of the respondents were presented with a series of options for getting real-time bus departure time information and asked if they thought that such information would be useful. Respondents were also asked to rank and identify the three information sources they thought were most useful. We found the following:

- 71 percent of the regular Transit Watch users (and 63 percent of the occasional users and 49 percent of the non-users) said that video monitors that provided bus departure time information in nearby shopping malls and the lobbies of major buildings would be useful.
- About half (55 percent of the regular Transit Watch users, 50 percent of the occasional users, and 59 percent of the non-users) said that having a large changeable sign outside the Transit Centers that could be read from the street would be useful.
- 70 percent of both the regular and occasional Transit Watch users (and 55 percent of the non-users) said more monitors at Transit Centers would be useful.
- About two-thirds of the regular Transit Watch users (as well as 55 percent of the occasional users and 45 percent of the non-users) said an Internet Web site providing bus departure information would be useful.[5] Phone lines providing this information were slightly less popular.

[4] The user group, composed of regular and occasional users of Transit Watch, comprises 50 percent of the sample, and the balance of the respondents are combined into a non-user group for this analysis. Percentages may not sum to 100 percent due to rounding.

2.3.3 Satisfaction, Value and Behavioral Responses of Transit Riders Based on Use of Transit Watch and Access to Real-time Bus Status Information

In general, users seem to be satisfied with the content, accuracy, presentation, and location of Transit Watch information. Both regular and occasional Transit Watch users view Transit Watch as a real benefit – more than a cosmetic addition to the Transit Centers. However, the responses also suggest that although Transit Watch seems to provide a variety of benefits to many users – some peace of mind and some flexibility – it does not significantly increase their satisfaction with their decision to use the bus. However, use of Transit Watch has a measurable effect on the comfort and satisfaction of new riders with the transit experience, and this has the potential to help retain ridership.

- Over three-quarters of the regular users indicated that information about actual and scheduled departure times, and the description of different route numbers was "very useful."

- Eighty-six percent of the users thought that the screen was very readable at close proximity, and over 90 percent thought that the locations of the screens were acceptable ("good" or "OK").

2.3.4 Behavioral Responses

The survey asked users about their reactions to learning about cancelled or delayed bus service (over five minutes) from Transit Watch. Users were also prompted for a wide range of possible behavioral responses and asked if they had ever reacted to late/cancelled bus information from Transit Watch with each of those responses. The analysis of the responses to these questions suggests the following:

- About three-quarters of the users of Transit Watch recalled at least one occasion when the Transit Watch video monitors informed them of serious delays (more than 5 minutes). Forty percent of these respondents agreed that in such situations, the information from the video monitors made them *less worried*.

- Moreover, a significant number of users indicated that they had responded with actions (e.g., calling home, taking different buses, driving home, etc.) to cancelled or late bus information provided by the video monitors.

2.3.5 Potential Influence of Transit Watch on Ridership

The results suggest that use of Transit Watch does have a measurable effect on the comfort and satisfaction of new riders with the transit experience. While we can't really say with these data whether a decision to implement Transit Watch at many other transit centers in the Metro system will retain more riders, increase their intensity of use of transit, or perhaps even attract new riders to the system, we can say that Transit Watch has had a positive effect on selected groups of transit riders who are traditionally more difficult to attract and retain. New frequent transit riders report the most satisfaction with their decision to take the bus since the introduction of Transit Watch. There is also some evidence here that some of these new riders are inclined to stay with transit, even

---

[5] Since this survey was completed, King County Metro has implemented on a trial basis for selected bus stops an Internet version of Transit Watch called MyBus. This new application will be evaluated separately outside of the MMDI program.

[6] See Charles River Associates, *Trends in Single Occupant Vehicle Miles and Miles of Travel Growth in the United States*, Final Report, 1998, published as "Web Document 5" by the Transit Cooperative Research Program and available at the National Academy Press Web site at www.nap.edu. This article includes a discussion of the determinants of transit ridership and the role that policy can play to influence ridership. Also, see Northwest Research Group, Inc., *1998 Rider / Nonrider Survey*, a report prepared for the King County Department of Transportation, Transit Division. This report identifies several key factors affecting ridership, including direct service to riders' destinations, more direct runs without a need to transfer, and service frequency, especially to work sites

when they do have options. However, bus information, such as that provided by Transit Watch, is only one factor out of many that can impact ridership.[6]

2.4 Summary and Discussion of Major Findings

The Major Findings of this analysis are summarized and discussed below:

- Transit Watch is both widely used and useful. Actual bus departure times are the Transit Watch feature found most useful by the users.
- Real-time information at locations where key travel decision are made (e.g., office buildings) would be used and considered useful by a majority of transit passengers. Transit Watch users particularly endorsed this suggestion.
- The content, location, accuracy, and presentation of the current Transit Watch monitors are satisfactory for most transit riders who use them, though many also offered suggestions for improvements.
- Although Transit Watch and the improved information *is* perceived as a real benefit by its users, the users did not seem to think that it increased their *overall* satisfaction with the transit experience. Our analysis indicates that Transit Watch in and of itself is unlikely to significantly change aggregate transit trends and perceptions. However, use of Transit Watch has a measurable effect on the comfort and satisfaction of new riders with the transit experience, and this has the potential to help retain ridership.
- Real-time information at locations where key travel decisions are made would be used and considered useful by a majority of passengers. Consumers expressed interest in a variety of forms other than Transit Watch monitors that would allow them access to real-time traffic information. The responses indicate that easy access to real-time information provided at *key decision points* would be useful. Consumer also suggested installation of Transit Watch-type video monitors at major bus stops. However, this finding may just reflect the important role of hands-on experience in framing consumer perceptions. Apart from video monitors at major bus stops, the most worthwhile information related investments seemed to be:
  - *Internet Web sites* that consumers could check before leaving work or home
  - Video monitors at *malls or office buildings close to major bus stops* that would allow consumers to maximize the time spent at their trip ends.

BusView

3. BusView [7]

3.1 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.

3.1.1 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 3-1, 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.

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

[7] Excerpt from final technical report, *Busview: An APTS Precursor and a Deployed Applet*, WA-RD 467.1, Research Project T9903, Task 43, June 2000. Authors: D.J. Dailey, S. Maclean, and I. Pao.

computer to keep up with the data flow, and therefor it does not coexist on a CPU with the other Operators in these examples.

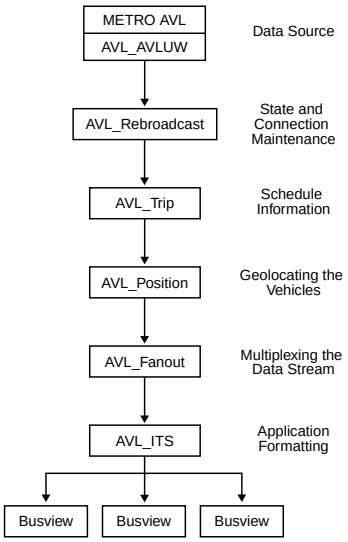


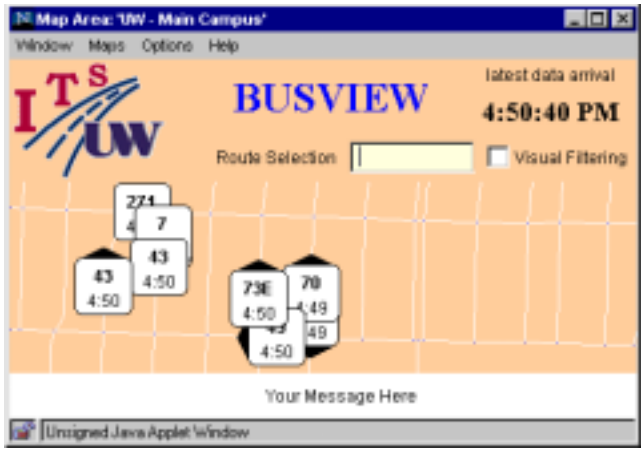
Figure 3-1. Data flow and architecture of Busview

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.

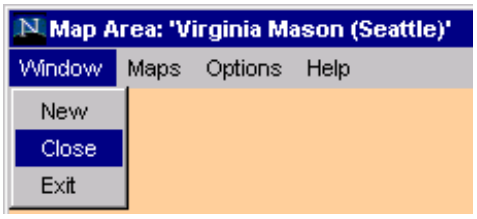
3.2 Graphical User Interface: Busview

3.2.1 Introduction



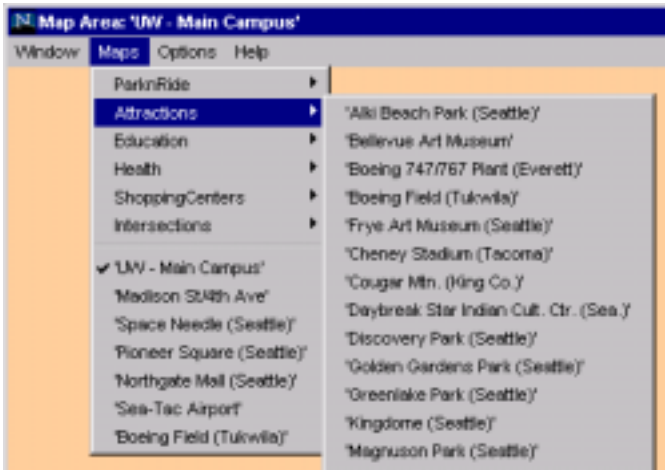
The image above is a generic screen shot of the Busview applet showing the default map of the University of Washington campus. Users can select another location from maps that cover the King County area. Each black and white icon on the map represents a different bus moving along its route. In the upper 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.

3.2.2 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.

3.2.3 Maps Menu

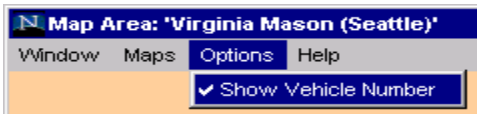


The Maps menu allows the user to cascade through a series of pull-down menus to select a region of King County one mile wide. Once the selection has been made, a new map is drawn. Buses appear as they move onto the map being viewed. A user can have multiple windows open for various locations by choosing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

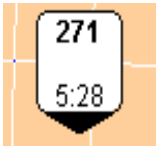
The default map selection is for the University of Washington campus in Seattle. Several maps have been pre-loaded for easy selection. A history list remembers the last seven maps that were viewed. When a new map is displayed, it moves to the top of the history list. The current map has a check mark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

3.2.4 Options Menu

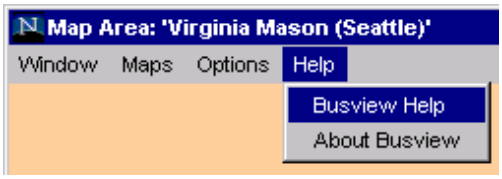


Currently, only one option is available under this menu. nly one option is available under this menu. This option toggles the vehicle identification number (VIN) on and off for the bus location icon. The VIN is a number that is painted on the back and top of each bus. As other options are available,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



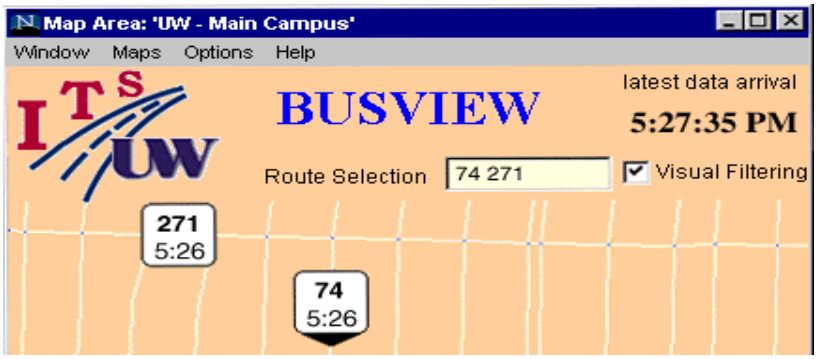
The image on the left shows the bus location icon with the VIN on, and the second image shows it with the VIN off. The default for the main maps is for the VIN to be off. When the bus progress bar is being viewed, the VIN is always on. This allows the user to track the correct bus along an entire route.

3.2.5 Help Menu



Choosing "Busview Help" links the user's browser to this Web page. Choosing "About Busview" displays the Busview version number.

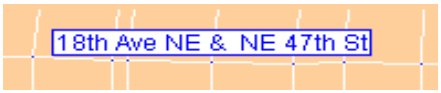
3.2.6 Route Selection/Visual Filtering



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.

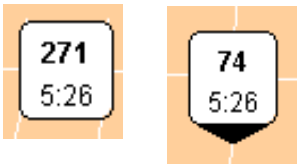
This may produce a blank screen if no buses are running on the selected route(s). Once buses from the selected route(s) enter the map area, however, they will be visible.

3.2.7 Street Names



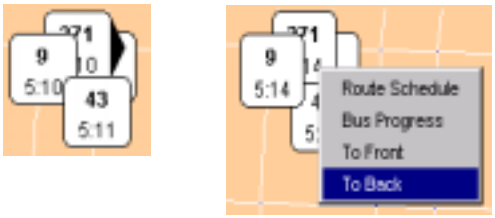
Intersections on the map have blue "hot spots." When the cursor passes over them, a street name pops up, as seen in the above image, .

3.2.8 Bus Location Icon



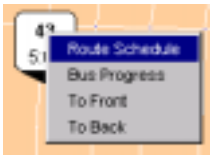
These icons represent the buses and appear on all map screens. The top number (in bold type) is the route number of the bus. The bottom number is a time stamp showing the last time that the bus was heard from. This time stamp should be compared with the "latest data arrival" time display in the upper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



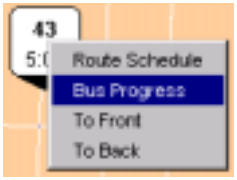
At times, some bus icons may be partially hidden because of the number of buses showing on the screen. Clicking on a bus location icon will bring up four additional options. The last two options, "To Front" and "To Back," allow the user to choose which icon to view. In the images above, clicking on "To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

3.2.9 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.

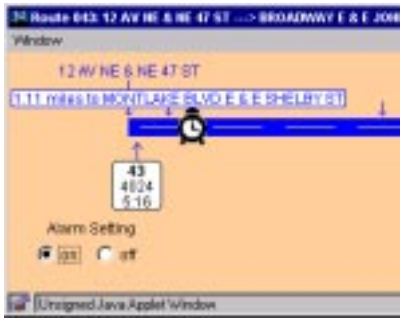
3.2.10 Bus Progress



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.

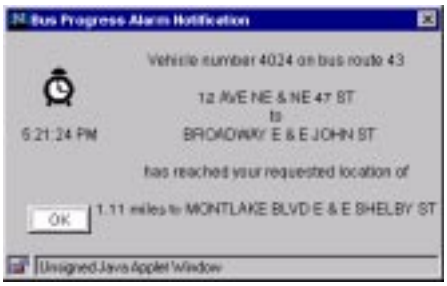
Buses move from left to right on the screen. The beginning and ending points for the route are listed both at the ends of the blue “roadway” and at the top of the window bar.

3.2.11 Alarm Setting



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.



3.2.12 Latest Data Arrival Time Display

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 (e.g., 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.

3.3 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 (i.e., 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

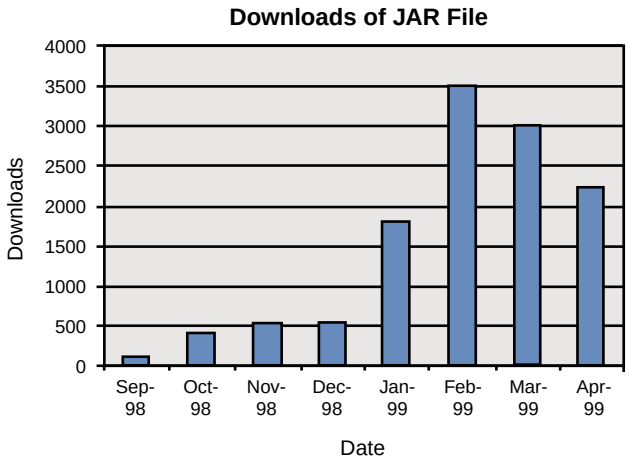


Figure 3-2. JAR downloads by months

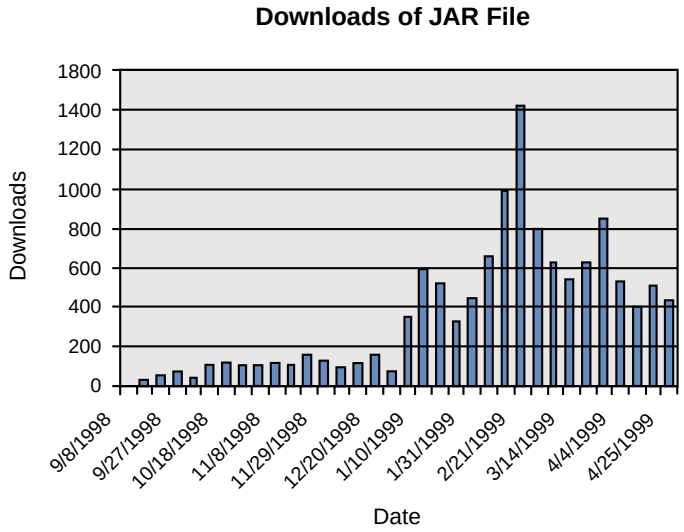


Figure 3-3. Downloads by weeks

two subsections below present the statistics for these two activity measures from intial deployment of the Busview applet to the date of this writing.

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 3-2, the downloads as a function of month are shown. Figure 3-3 shows the number of downloads each week over the 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 3-4 shows the use by day of week, indicating that the applet is more heavily used during the work week than on weekends. Figure 3-5 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

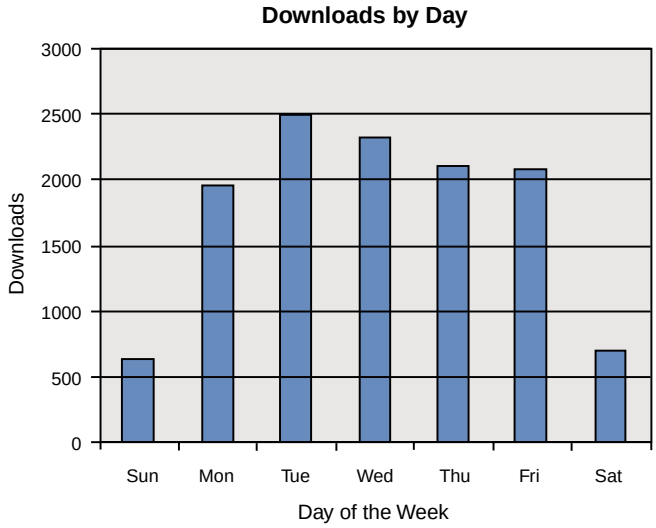


Figure 3-4. Downloads by day

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.

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 3-6 shows the number of uses each week over the reporting period. Figure 3-7 shows the number uses each day for the reporting period. Figure 3-8 shows the use by day of week, indicating that the applet is more heavily used during the work week than on weekends. Figure 3-9 shows the data stream usage by time of day, and 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

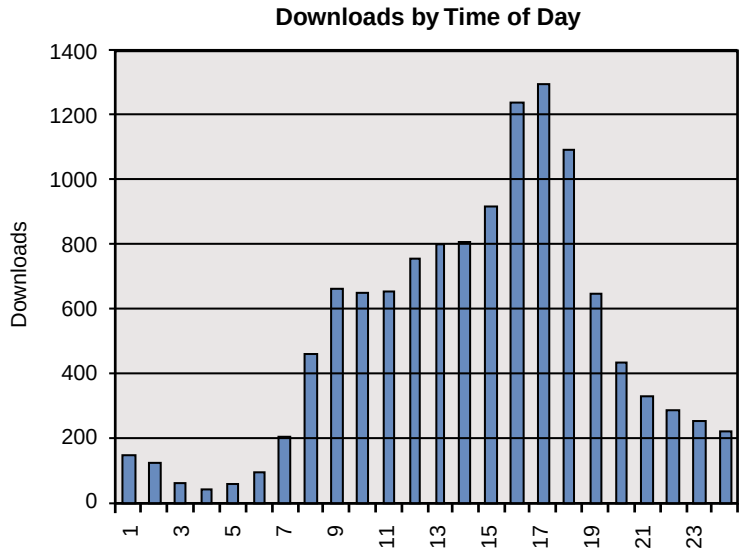


Figure 3-5. Downloads by time of day

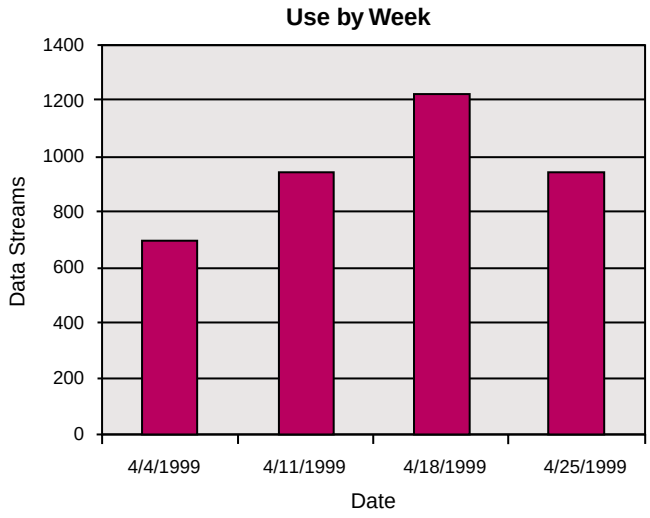


Figure 3-6. Data stream connection by week

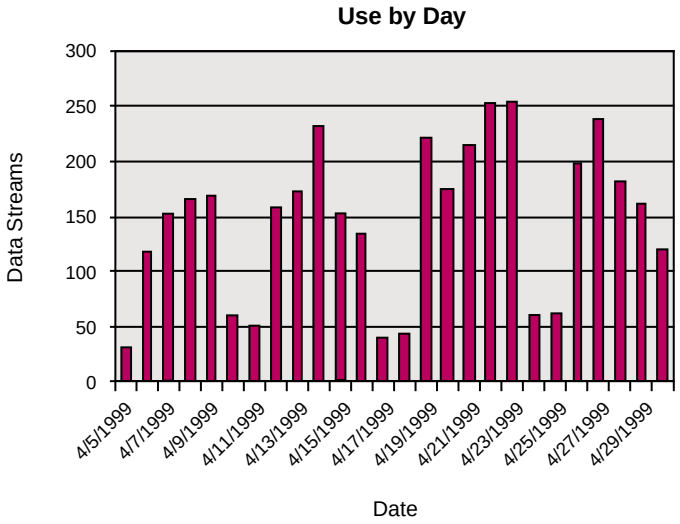


Figure 3-7. Data stream connections by day

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 Mililtary)
9	.au (Australia)
7	.be (Belgium)
7	.hk (Hong Kong)
6	.sg (Singapore)
5	.fi (Finland)
5	.it (Italy)
4	.es (Spain)
4	.dk (Denmark)
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)

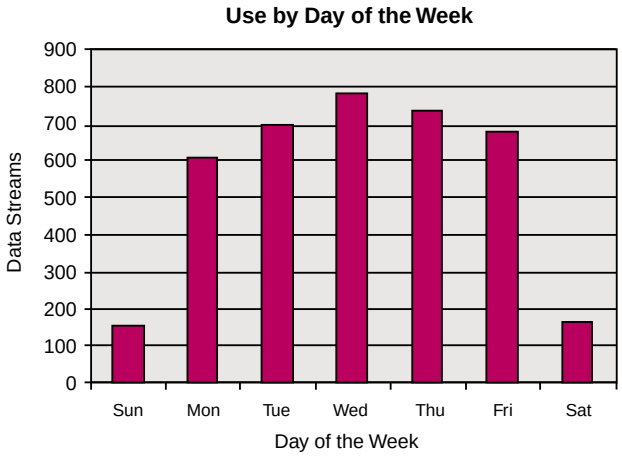


Figure 3-8. Data stream connection by day of the week

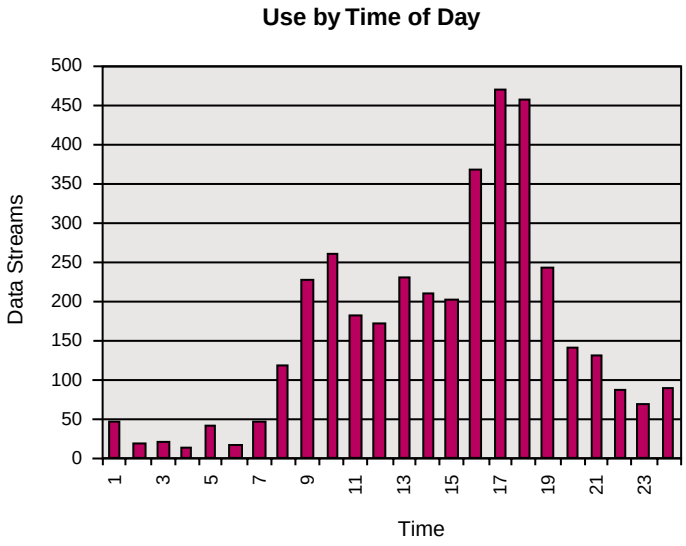


Figure 3-9. Data stream connection by time of day

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 numberical 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)

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.

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 below. In general, Busview was well received, and several useful suggestions for improvements were included in the comments.

3.4.1 Busview Comments

Dear busview team,
I think busview is really cool. It has greatly improved since the version that I saw a few months ago.

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

This is the BEST applet EVER!!!!

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.

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.

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.

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!

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.

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.

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-)

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.

Hello and congratulations on a great application! My life is much easier since I discovered Busview. Thanks for your help.

4. BusView Focus Group Report [8]

4.1 Introduction

Busview is a free internet website with graphical representations of King County Metro buses as they move through the County. A square icon tracks each bus on every Metro bus route. The icon moves along a street grid, stopping at intersections and bus stops as it travels along its route. A series of maps now cover the County. Users may select to view a single route or number of routes in one or both directions. In Phase II, BusView users will be able to "tag" a particular route, location, and time so that they are alerted by a signal when their bus passes a pre-determined geographic location.

In an effort to identify the thoughts, feelings and behaviors of BusView users, a focus group was conducted. Specifically, the focus group was designed to address issues around:

- How BusView affects use of King County Metro
- Reactions to the current BusView prototype site
- Reactions to anticipated additional BusView features
- Reactions to the potential to have BusView or other real-time bus information available at retail establishments

4.2 Methodology

Focus groups involve a discussion with a small group of people on a specific topic. Groups typically involve from six to ten people and last from one-half to two hours. Participants are typically homogeneous in characteristics central to the focus group topic and purpose. The major benefit of focus groups is that participants are able to hear each other's responses and to make additional comments beyond their own original responses. This synergy allows for insights to be reached that would ordinarily be missed with questionnaire or personal interview techniques. With skillful focus group facilitation one is able to go beyond the *superficial* level of respondents' answers and reach the *authentic* and *core values* levels.

Recruitment was conducted by Pacific Rim Resources (PRR) after obtaining a list of people who responded to a solicitation for focus group participants on the BusView website. Potential participants were screened in regard to the following criteria:

- Must have used BusView within the last month (following some site improvements)
- Equal distribution of *regular* (3 out of 5 working days per week) and *irregular* (less than 3 out of 5 working days per week) users of King County Metro
- Must have been 21 years of age or older
- Between one-third and one-half needed to be from the eastside of Lake Washington
- Equal distribution of males and females

Of the twenty-three people who responded to the solicitation for participation in a focus group, PRR over-recruited ten-for-eight participants. Due in part to a $50.00 incentive and close contact with those that had agreed to participate, eight participants showed up. These eight participants had the following characteristics:

- Gender unbalanced: 6 males and 2 females

[8] Prepared by Bruce Brown, Ph.D., Research Director, Pacific Rim Resources, February 9, 1999

- Five from the westside and three from the eastside of Lake Washington
- Most are regular Metro users, with only one person indicating that they use Metro less than 3 out of 5 working days
- Access to the internet occurs for five participants at both home and work, for one person just at home, for another just at work , and finally for one person at school or a public computer lab
- Three participants currently have a mobile computer device (either Palm Top or Palm Pilot), but only one of whom has Microsoft CE installed. Another participant used to have a Palm Top, but no longer does so.
- Other mobile communications devices included four people who have cell phones and four people who have pagers. Three of these people had both a cell phone and a pager.

The focus group was held on the evening of January 21, 1999 at the focus group facilities of Pacific Rim Resources and lasted approximately two hours. Bruce Brown (Research Director at PRR) and Michael Richards (PRR consultant to SmartTrek) moderated the focus group. For purposes of analysis, the group was videotaped, audiotaped, and unobtrusively observed by members of the BusView website design team or SmartTrek Program. The moderators used a discussion guide to focus on issues of importance regarding BusView.

4.3 Findings

4.3.1 Current Metro Transit and BusView Use

1. Most participants indicated that they use Metro on a daily basis for a variety of purposes including:
   - Commuting to work
   - Commuting to the university
   - Whenever coming into downtown Seattle for entertainment, visiting friends, or shopping

   Less frequent uses of Metro include traveling to the airport and for special events where parking would be a major difficulty.

   In addition to the participants' use of Metro, they report a fairly frequent use of Metro by other family members for similar purposes. This fairly high use of Metro is not due to the fact that these families do not own cars (only two people do not own cars), but rather because they find Metro convenient and an environmentally responsible alternative.
2. Participants mentioned a number purposes for which they use BusView including:
   - Useful because their bus is typically late once every third or fourth day
   - Useful when buses are delayed due to inclement weather
   - The route progress feature is particularly useful when using a route other than one's typical route
   - More useful for areas outside of downtown since there are so many buses available in downtown
   - Would be useful to have on a mobile service for routes when transferring is involved
3. Regarding how they use BusView, there were a variety of responses including:
   - Pulling up BusView about fifteen minutes before leaving for their bus

- Leaving BusView on monitor overnight so that it is available first thing in the morning
- Checking Busview just before leaving for work in the morning and just before leaving work in the evening

4. About half of the respondents agreed that BusView has increased their use of Metro, while the others indicated that it has not had any impact since they are already committed to using the bus. In particular, BusView increases Metro use:
   - For destinations that the person typically does not travel
   - Where buses often do not run on time
   - Where one does not know the route schedule
   - For routes where the bus stop may be a safety concern during certain times of the day
5. When asked how they would explain BusView to other people unfamiliar with the site, they mentioned the following:
   - "It let's you see where your bus is."
   - "It let's people know if they have more time before having to catch the bus."
   - "It's a cool thing."
6. Participants found out about BusView from the following sources:
   - SmartTrek ads on bus boards
   - Washington State Department of Transportation website
   - While surfing the web
   - From a former Metro focus group
7. Participants suggested the following methods for other people to learn about BusView:
   - Have staff at park & rides and bus stops demonstrating BusView
   - Advertise on bus boards and inside buses
   - Include information about BusView on bus schedule time tables
   - Have a link to BusView on the Metro website

4.3.2 Reactions to BusView

4.3.2.1 Reactions to the BusView interface

When questioned about the current BusView prototype there was general agreement that although the site is helpful and based on a useful concept, that a number of improvements are needed. The following are the concerns raised by the participants:

- Additional cues are needed to provide first-time users with information on how to use BusView
- Depending on the type of operating system one is using, BusView can be very slow at times. This was particularly the case for opening the specific maps that one is interested in viewing.

- The menu structure that currently exists is time-consuming and cumbersome to navigate through in order to reach the desired maps. ("Menus upon menus," as described by one participant.)
- The clock should be set to Metro's time, not the individual's computer clock
- The route progress feature would be more useful if it indicated the closest intersection to the buses present location, as opposed to what percent of the route is completed. When questioned about a preference for the *bird's eye view* or the *route progress view*, participants indicated no preference and stated that each was useful for different purposes. (The route progress feature was considered more useful for routes that one is not familiar with.)
- All agreed that knowing the estimated time of arrival at ones' bus stop would be a major improvement
- One respondent found that they can not see all of the intersection choices when attempting to select the appropriate map
- All agreed that the inability to bookmark one's map and specific routes of interest is a major drawback
- One person mentioned being able to click on the bus icons for their route choices is preferable to the route selection box as a method of filtering for routes
- There was a preference for the ability to zoom-in and zoom-out on the maps, a feature that does not currently exist
- Most participants would like to be able to view about two routes at a time, with no more than four at any one time
- The bus icons were viewed very favorably. They were seen as basic, but clean and easy to read. The bus number information, although not seen as of critical importance, gives the icon a more realistic look and balances the information within the icon. (One person mentioned that the directional arrows on the bus icon can give the impression of a three-dimensional object as opposed to indicating bus direction.)
- A suggestion was made to have icons on the downtown Seattle map showing points of interest. This would provide users with some landmarks to more easily orient themselves to the map and would have the added benefit of being useful to out of town visitors using BusView.
- Although participants liked the feature wherein street intersections appear when clicked on, they suggested that a few major streets should always be shown on the maps to help orient the user.
- There was no concern with the use of a tan background and white streets
- There needs to be a way to tell that buses are stacked on top of each other on the screen

4.3.2.2 Reactions to proposed new features

- When introduced to the possibility of BusView first appearing as a map of King County on which one could click the area where their bus stop was located in order to open a more detailed map of their area, the idea was met with universal approval. There was complete agreement that the current series of menus used to locate one's map was a less desirable approach, even if the menus were to function more quickly (which they are expected to do in about a month's time). During this discussion the idea of having a menu of neighborhoods to choose from was presented by one of the participants. This met with mixed interest.
- In regard to the ability to scroll from one map area to adjacent map areas, participants were very interested in such a feature since their routes often traverse more than one map. Other methods of moving from one map to another suggested by participants included being able to click on choice boxes that would move the screen, for example, up a half a screen, to the left one full screen, etc. The idea of

having directional arrows on the corners of the screen to take you to the next map was also acceptable, but did not address the issue of wanting to see one's route over two maps at the same time.

4.3.2.3 Participants' interest in being alerted that your bus is approaching

- Participants saw this as a very useful feature, but wanted to be able to deselect the feature as desired and wanted to be able to indicate the time periods during which they wanted to be alerted. There was general agreement that an audible alert was preferable to an icon appearing on one's computer screen. This preference was due to the fact that people aren't always looking at their computer screen, but could none-the-less hear an audible alert. Participants agreed that no more than four bus routes would need to be alarmed in this way at any time. In regard to how much in advance people would want to be alerted, that was determined to be user dependent, i.e., some people need 10 minutes to get to their bus stop, others need 15 minutes, and some will need 20 minutes to drive to a park and ride lot.
- Sending such alerts to one's pager, cell phone, or personal digital assistant was considered very useful by the participants. However, they made it very clear that such messages must be readable in full and not cut the message off as is currently the case with some county message systems. Several participants who currently do not use a pager indicated that they might buy or rent one if such alerts could be sent. They would definitely do so if the alerts could tell them the estimated time of arrival of their bus.
- In general there was great interest in the idea of being able to get bus information on mobile devices. Most participants were willing to pay $5.00 a month for such a service. Ten dollars a month was considered to be too expensive for such a basic service (although one participant was agreeable to this). Participants preferred a use-based fee over a flat fee for the mobile service. Finally, participants were not opposed to having advertising along with their bus information, in order to keep the cost of the service down, as long as the advertising did not interfere with the bus information.

4.3.3 Real Time Bus Information at Retail Establishments

Participants were in favor of providing real time bus information at retail establishments such as coffee shops, shopping centers, etc. The advantage to them would be that they would be able to continue their activity and still get to their bus stop on time.

Of the three choices of displaying the real time bus information, the favorite was a system similar to an airport style readerboard. This was chosen over displaying BusView because it would also provide information on estimated time of arrival. The option of an LED sign display scrolling through each route was uniformly rejected due to the fact that only a few routes could be viewed at a time thereby taking longer to get the information. Also, such displays were considered difficult to read. Participants urged that the airport style readerboard should continue to display buses that have already left so viewers would know that in fact they have missed a particular bus.

4.3.4 BusView Advertising in Metro Time Schedule Displays

Participants were shown two examples of advertisements for BusView that are designed to be placed in the long and narrow panels of Metro schedule displays. Several issues emerged in regard to the ads:

- Participants agreed that the "go to www.BusView.com" part of the message did not require the "go to" part. It was felt that people would understand that BusView would be available at the website address. Also, eliminating the "go to" section would allow the website address to be printed in larger print.
- The person in the ad pointing to the computer screen suggests that BusView operates in a "touch screen" manner. Since this is not the case it was suggested that the graphic be changed to correct this misperception.

- Participants were not overly impressed with either of the two tag lines connected with the ads (although they preferred “Know where your bus is before you leave to catch it” over Where’s my bus? Its online.”)
- Participants suggested the following tag lines:
  - “Where’s my bus right now? Find it online at www.BusView.com.”
  - “It’s 5:00 o’clock. Do you know where your bus is?”

4.4 Summary

A focus group of King County Metro and BusView users was conducted in order to assess the current BusView website, as well as planned future improvements to the site. Participants found BusView to be a useful tool, particularly in situations where their bus was expected to not run on schedule due to traffic or weather conditions. In addition, BusView was seen as useful for getting information regarding routes that they typically did not use.

Although a number of suggestions were made for improvements, four issues were identified as critical by participants:

- The current method of selecting one’s map of interest needs to be improved so that one does not have to sift through a series of menus
- The speed of particular areas of the site need to be increased; specifically the speed with which maps and bus icons appear
- The site needs a way to bookmark one’s map and bus route selections
- Information regarding the *estimated time of arrival* of one’s bus would be a major improvement

In regard to proposed new features, participants were unanimously in favor of the following:

- Having a feature that would alert one that their bus was “x” minutes away from their stop
- Having the ability to scroll from one map to adjacent maps
- Having BusView open up with a map of King County upon which one would select and click an area to open up more detailed maps
- Having the ability to view BusView on mobile devices and to be alerted on mobile devices (pager, cell phone, etc.) that one’s bus is approaching one’s bus stop
- Having the ability to view real-time bus information at retail establishments

5. BusView Usage Statistics

5.1 Web Usage Statistics for Busview.jar Downloads

In this section, we present some statistics obtained from the usage logs created by the Busview web server. These reflect a snapshot taken on February 15, 2001 for approximately 19-months of operation (887 days). At the time of writing, the up-to-date statistics can be found at http://trolley.its.washington.edu:8080/stats/.

These statistics reflect specific tasks performed during a Busview session. Busview is implemented as a downloadable applet stored in a ".jar" file. The Busview user first connects to the site busview.org. The user's browser then determines the date on the ".jar" file. Then, if the browser already has a cached copy of Busview, it launches the local applet using the "build" in Java virtual machine. If the user's browser does not have a cached copy of Busview or if the version available from the web site is newer due to a new release of the software, the browser downloads the ".jar" file. So a ".jar" download reflects: (1) a new user, (2) a user with an old copy of the ".jar" file, or (3) a user with no browser cache. Figure 5-1 shows the number of downloads of the ".jar" file per week over the 19-month period. The total number of downloads is 131,111.

The overall trend is an increase of usage over time with fluctuations in the general trend. For example, the application received some online web publicity with developers in November of 1999, and there is a peak in the downloads reflecting this short-term publicity.

Figure 5-2 shows the number of downloads by day of the week is. The weekday usage is about four times that of weekend usage. The speculation is that people may use it to assist with their commute.

Figure 5-3 shows the number of downloads by hour of the day. There are peaks at 8-11 a.m .and 4-6 p.m., again suggesting that the application is being used in support of commuting activities.

Figure 5-4 shows the number of downloads by specific domain name. The top domains are either internet service providers, such as USWest, Northwest Link, or AOL, or they are large employers, such as Microsoft, Boeing or Amazon.com.

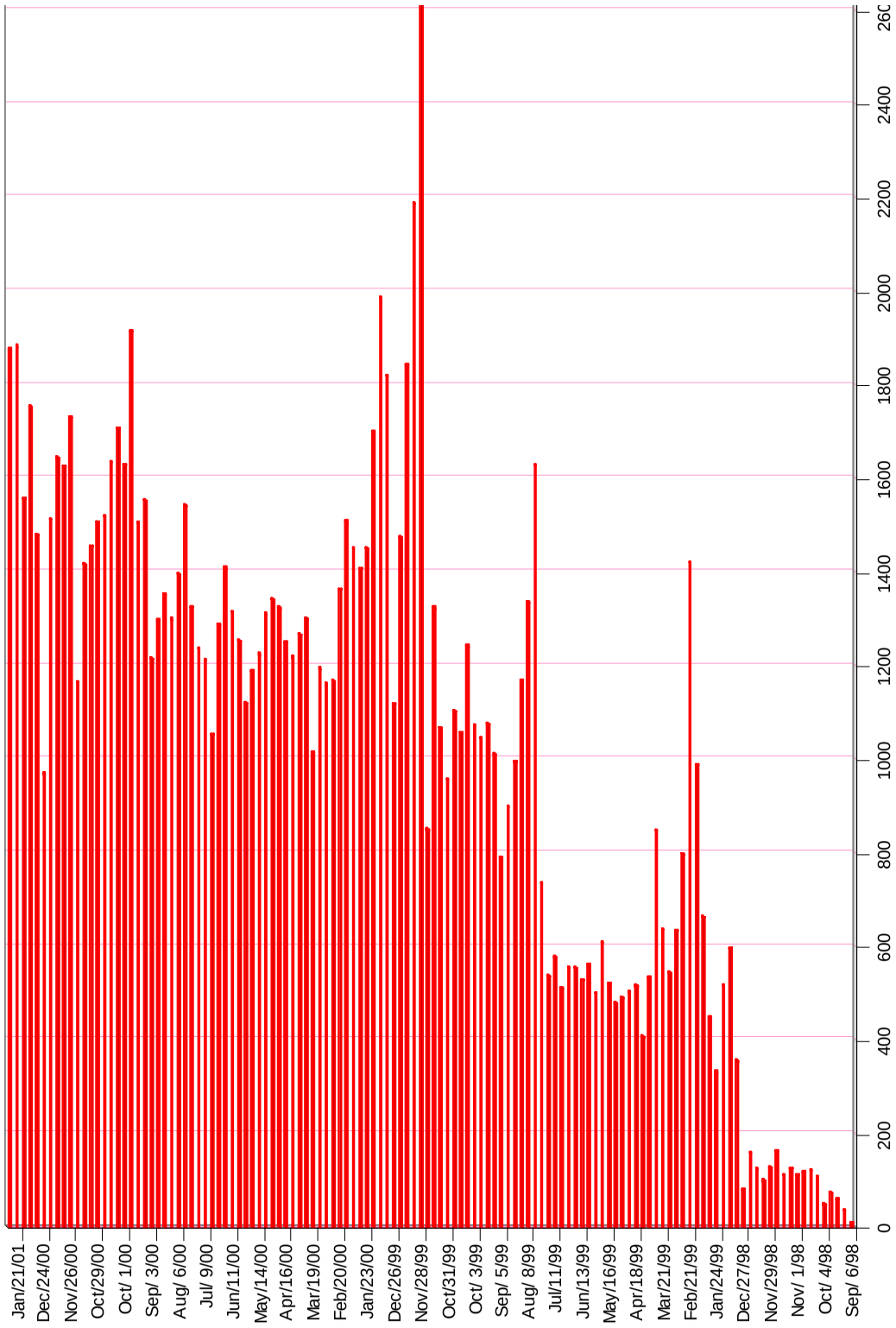


Figure 5-1. Busview.jar downloads weekly report

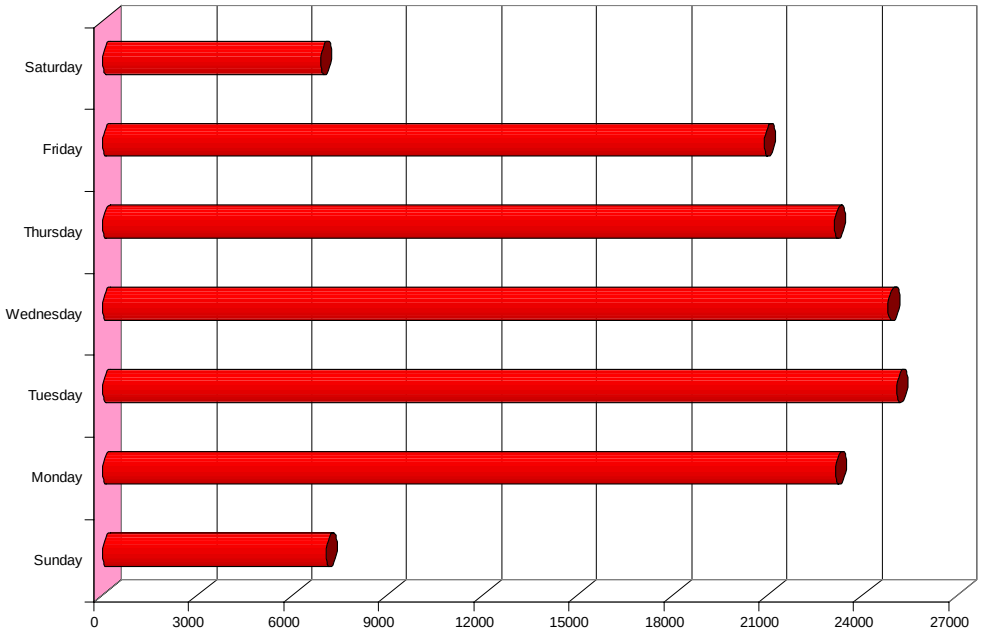


Figure 5-2. Busview.jar downloads daily summary

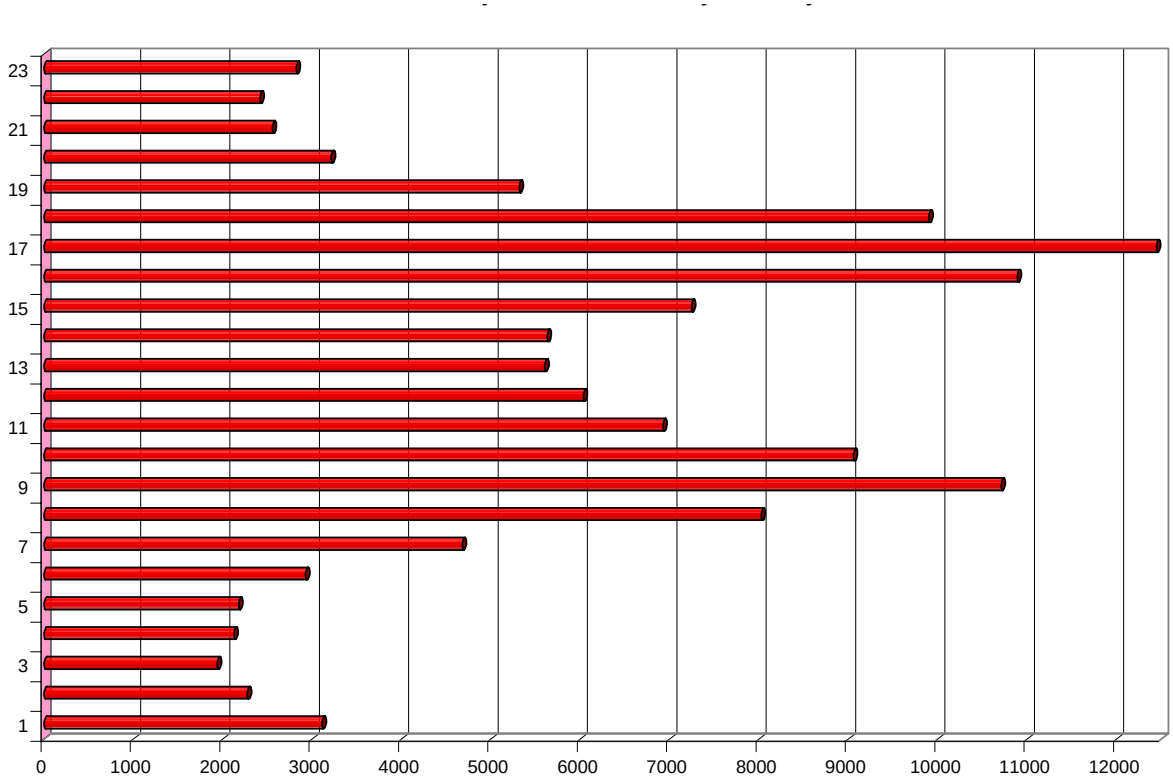


Figure 5-3. Busview.jar downloads hourly summary

Requests	Host
6,657	.uswest.net
6,260	.microsoft.com
2,735	.aol.com
1,872	.nwlink.com
1,853	.boeing.com
1,522	208-216-180-101.amazon.com
897	global.ci.seattle.wa.us
666	proxy2-external.sttln1.wa.home.com
558	king1.wsdot.wa.gov
540	chameleon.cobaltgroup.com
538	mulder.f5.com
530	elmo.nordstrom.com
499	proxy2-external.sttls1.wa.home.com
462	c693749-a.sttln1.wa.home.com
447	bullwinkle.korry.com
414	206.169.238.34
382	proxy1-external.sttln1.wa.home.com
374	c204-250-145-217.seattlelab.com
369	proxy.amc.com
359	c1358492-a.sttln1.wa.home.com
355	frisco.ch2m.com
344	spxylax1.bankofamerica.com
333	c620265-a.sttln1.wa.home.com
308	seafw01.primus.com
305	cache.airborne.com
301	roland.avenuea.com
297	t1.immunex.com
291	wakko2.prognet.com
282	user.onvia.com
276	jarvikj262.genetics.washington.edu
100,085	[not listed 33,534 hosts]

Figure 5-4. Busview.jar host requests

5.2 Web Server Statistics for Busview Data Streams

A second specific task in the usage of Busview is the initiation of the real-time data stream. After the Java applet is started in the local Java virtual machine, the applet connects to the server to obtain a continuous stream of real-time bus data. This connection remains in place throughout an entire Busview session and is only disconnected when the user closes the Busview application. As such, this statistic represents the best estimate of actual use of the application. The total usage is 185,732.

Figure 5-5 shows the number of uses per week over the 19-month period. Again, the trend is slowly upward with the November 1999 usage peak contemporaneous with the web publicity.

Figure 5-6 shows the number of downloads by day of the week. The weekday usage is about three times that of weekend usage. The speculation is that people may use it to assist with their commute.

Figure 5-7 shows the number of downloads by hour of the day. There is a small peak at 8-11 a.m. and a much larger one from 4-6 p.m. This suggests that the application is used in support of commuting activities and that users are more likely to request the information from their place of employment than from their home.

Figure 5-8 shows the number of downloads by specific domain name. The top domains are either internet service providers, such as USWest, Northwest Link, or AOL, or they are large employers, such as Microsoft, Boeing or Amazon.com.

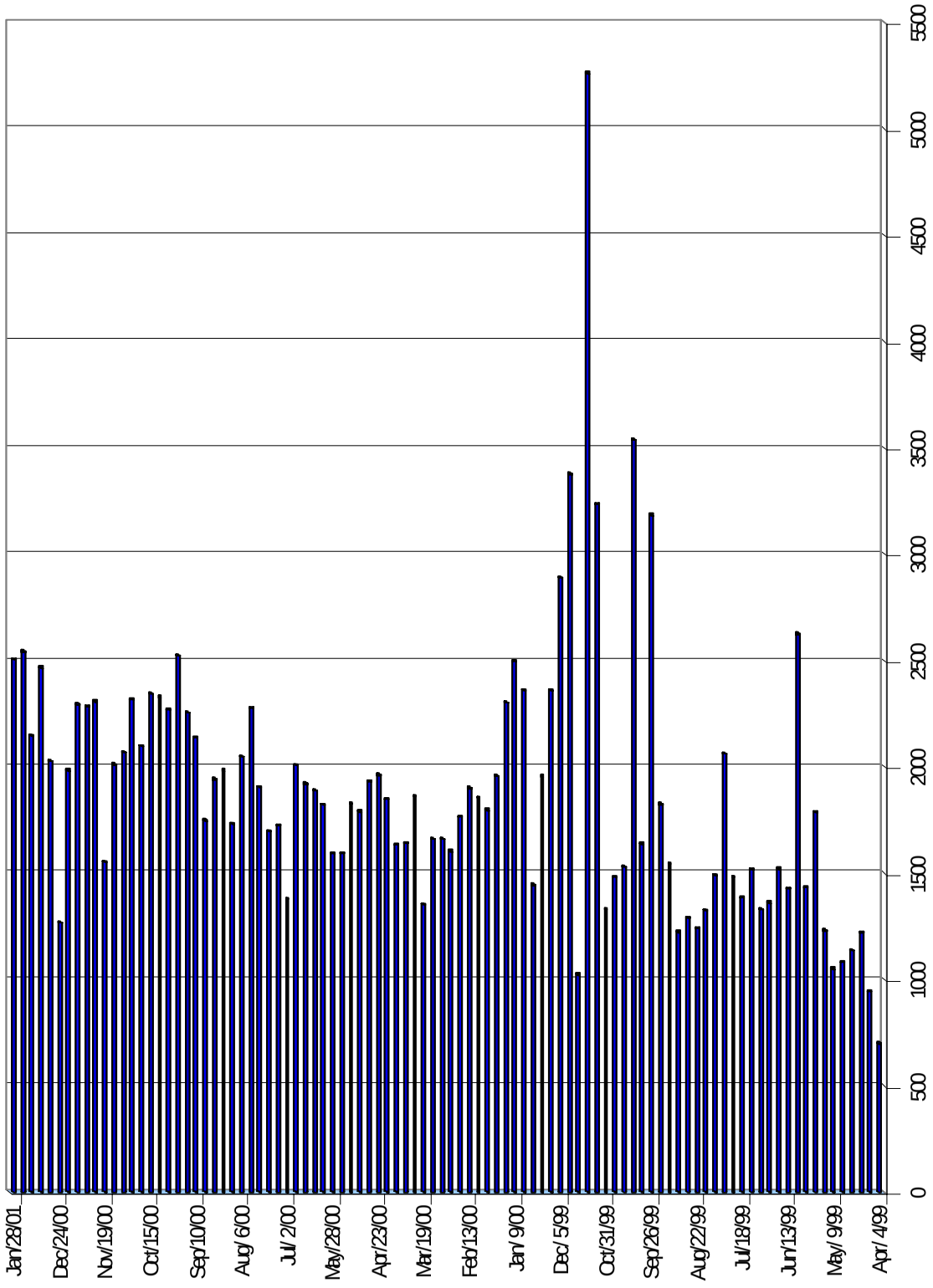


Figure 5-5. Busview data stream weekly report

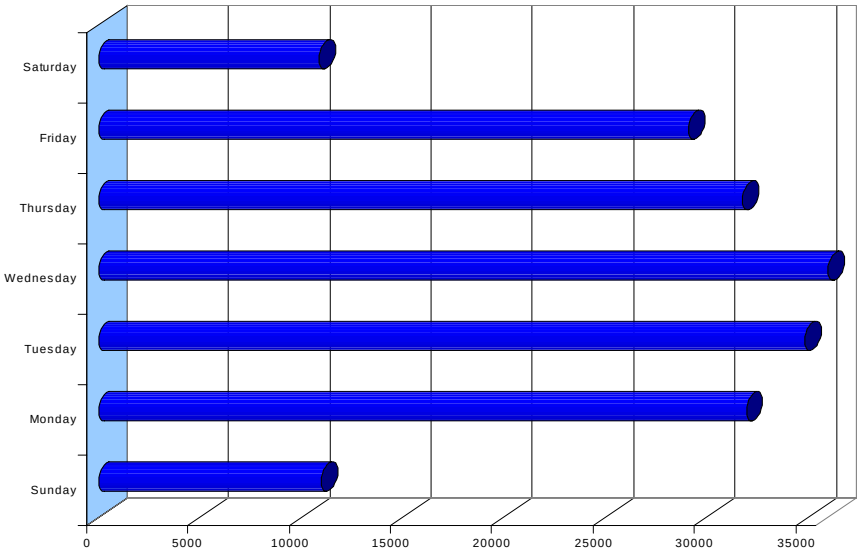


Figure 5-6. Busview data stream daily summary

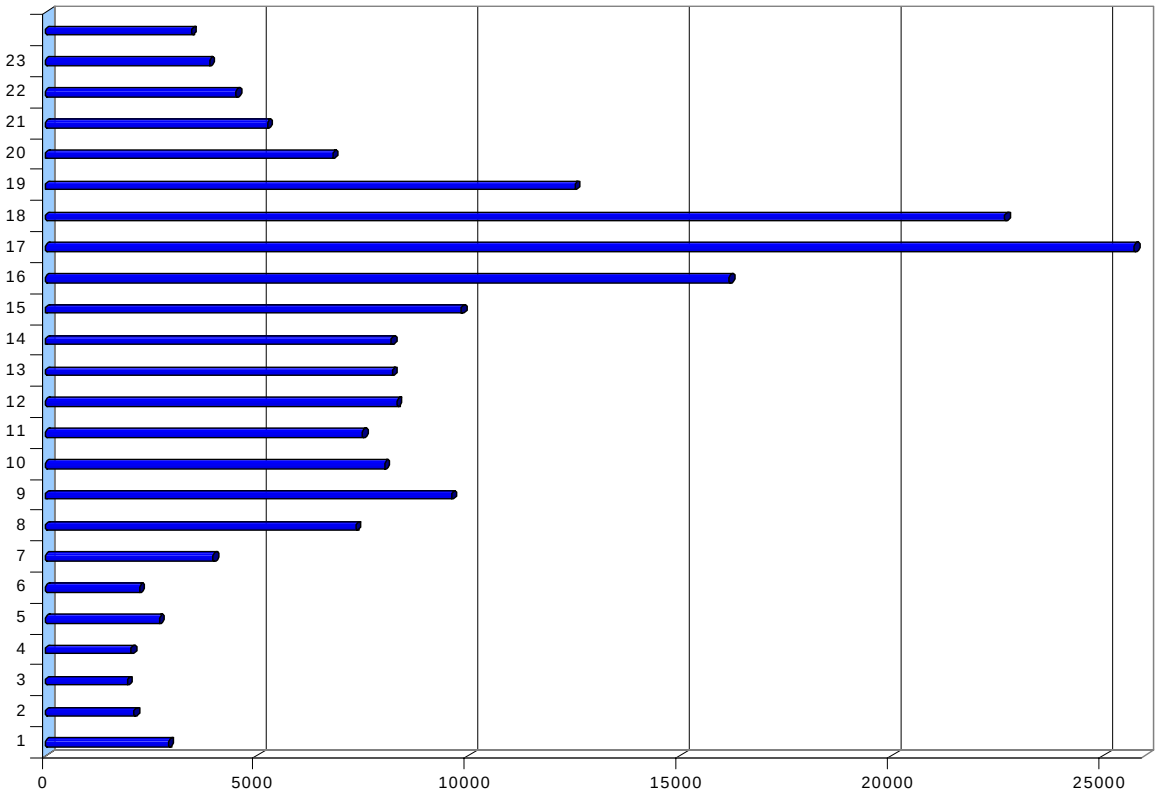


Figure 5-7. Busview data stream hourly summary

Requests	Host
37,108	.microsoft.com
7,730	.uswest.net
4,235	.aol.com
2,026	.nwlink.com
2,025	proxy.amc.com
1,888	208-216-180-101.amazon.com
1,410	global.ci.seattle.wa.us
1,023	proxy2-external.sttln1.wa.home.com
845	king1.wsdot.wa.gov
843	proxy1-external.sttln1.wa.home.com
806	proxy2-external.sttls1.wa.home.com
706	mulder.f5.com
637	chameleon.cobaltgroup.com
600	206.169.238.34
556	elmo.nordstrom.com
539	firewall.bsquare.com
527	spxylax1.bankofamerica.com
525	204.57.230.116
516	c693749-a.sttln1.wa.home.com
480	c1358492-a.sttln1.wa.home.com
440	sense-sea-cacheraq.oz.net
429	t1.immunex.com
419	bullwinkle.korry.com
378	c204-250-145-217.seattlelab.com
363	cache.airborne.com
118,678	[29,424 hosts]

Figure 5-8. Busview data stream host requests

Traffic Channel

6. Traffic Channel

One component of the Smart Trek project was the development of a professional quality traffic information channel for broadcast via television. The final version included traffic and traveler information such as real-time video images from surveillance cameras. The programming was initially deployed on the Univeristy of Washington cable television channel effective June 1, 1998 but is portable to other TV stations.

6.1 Overview of the Traffic Channel

The Traffic Channel is a mixed-media system designed to provide a highly-configurable system for a professional quality information channel. The stand-alone, PC-based system integrates real-time video with real-time data to provide a broadcast-ready, NTSC compatible output signal.

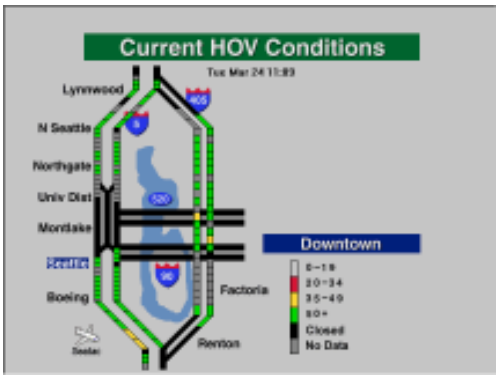
The primary Traffic Channel program, TrafficChannel.exe, is designed to have a broad range of display capabilities through end-user configurable files (Section 6.5).

This document describes Traffic Channel as configured for the UWTV implementation. This implementation is a thirty-second sequence that displays four maps for a period of 7.5 seconds each. The sequence starts at the 0- and 30-second marks of each minute and repeats indefinitely until the program is terminated by hitting the End key. The Genie Digital Video Effects board is used to perform video *page roles* between each map transition. *Swooshes* are used to bring traffic video images acquired from roadside cameras on to the presentation screen.

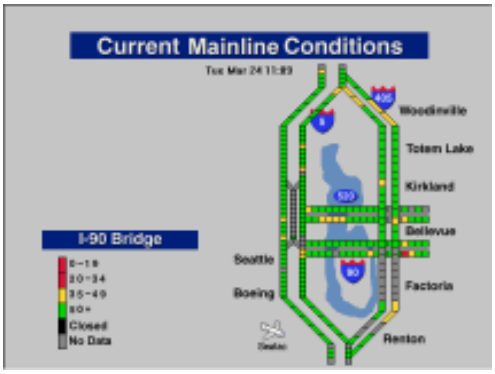
The four map/video selections for the UWTV implemenation are:

1. *Mainline Traffic - Route 520 bridge midspan:* map of mainline traffic data and video display of 520 bridge.

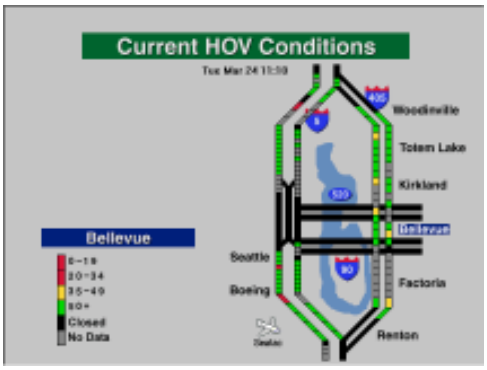
2. *HOV Traffic - Downtown Seattle:* map of HOV traffic data and video display of downtown Seattle.



3. *Mainline Traffic - Interstate 90 bridge midspan:* map of mainline traffic data and video display of I90 bridge.



4. *HOV Traffic - Bellevue NE 8th:* map of HOV traffic data and video display of Bellevue.



The Traffic Channel system relies on accurate timing for all events on the timelines shown in Figure 6-1.

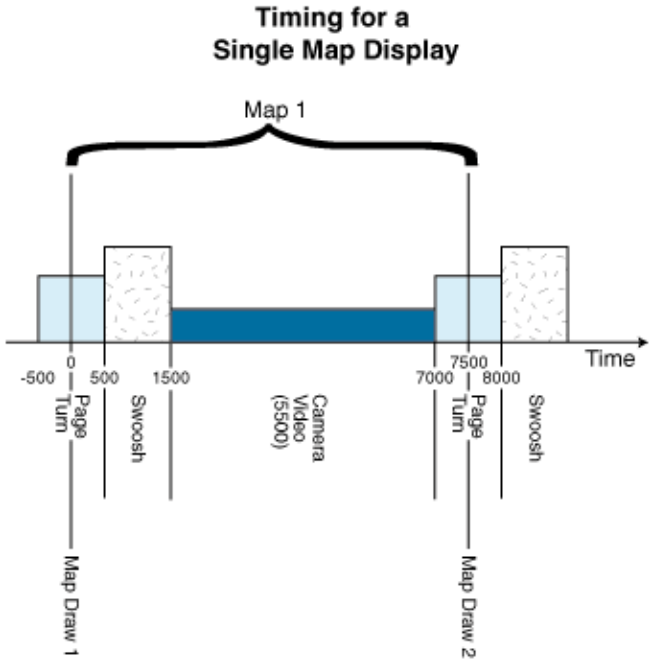
Television broadcast systems require timing accuracy to the second, and Traffic Channel must maintain even higher accuracy to be synchronized with the broadcast system. The presented video sequence (Section 6.3.2.1), which is controlled by the Video-Control Computer, must be closely synchronized with the video effects on the Display Computer.

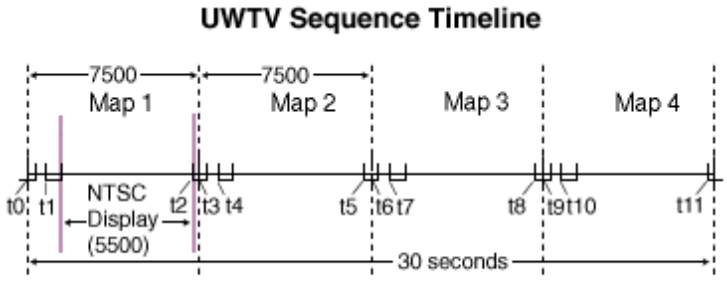
Operation of the UWTV implementation of the Traffic Channel requires the use of several computer and network assets both at and between the UW and WSDOT regional headquarters in north Seattle (http://www.wsdot.wa.gov/regions/northwest/, Michael Forbis, (206) 440-4475). Figure 6-2 illustrates the layout of these components. (See the Traffic Channel installation instructions, Section 6.3, in this document for installing and configuring the computer components and software.)

Note that the operation of the Video Control Computer and the Video Display Computer are tightly coupled in drawing the maps and controlling the video display, and both machines must be configured to operate in unison.

6.2 Traffic Channel System Requirements

These specifications are limited to the computer components of the TrafficChannel system. The complete broadcast-ready system requires other components, such as the live video feed and ITS backbone data (http://www.its.washington.edu/bbone/), which are not covered in this document.





Map 1: Mainline Traffic Data - 520 Bridge Video
Map 2: HOV Traffic Data - Downtown Video
Map 3: Mainline Traffic Data - I90 Bridge Video
Map 4: HOV Traffic Data - Downtown Bellevue Video

	Event	Start Time	Duration
to	Draw Map 1	0	7500
t1	Swoosh DVE	500	1000
t2	Page Turn DVE	7000	1000
t3	Draw Map 2	7500	7500
t4	Swoosh DVE	8000	1000
t5	Page Turn DEV	14500	1000
t6	Draw Map 3	15000	7500
t7	Swoosh DVE	15500	1000
t8	Page Turn DVE	22000	1000
t9	Draw Map 4	22500	7500
t10	Swoosh DVE	23000	1000
t11	Page Turn DVE	29500	1000

Figure 6-1. Traffic Channel timelines

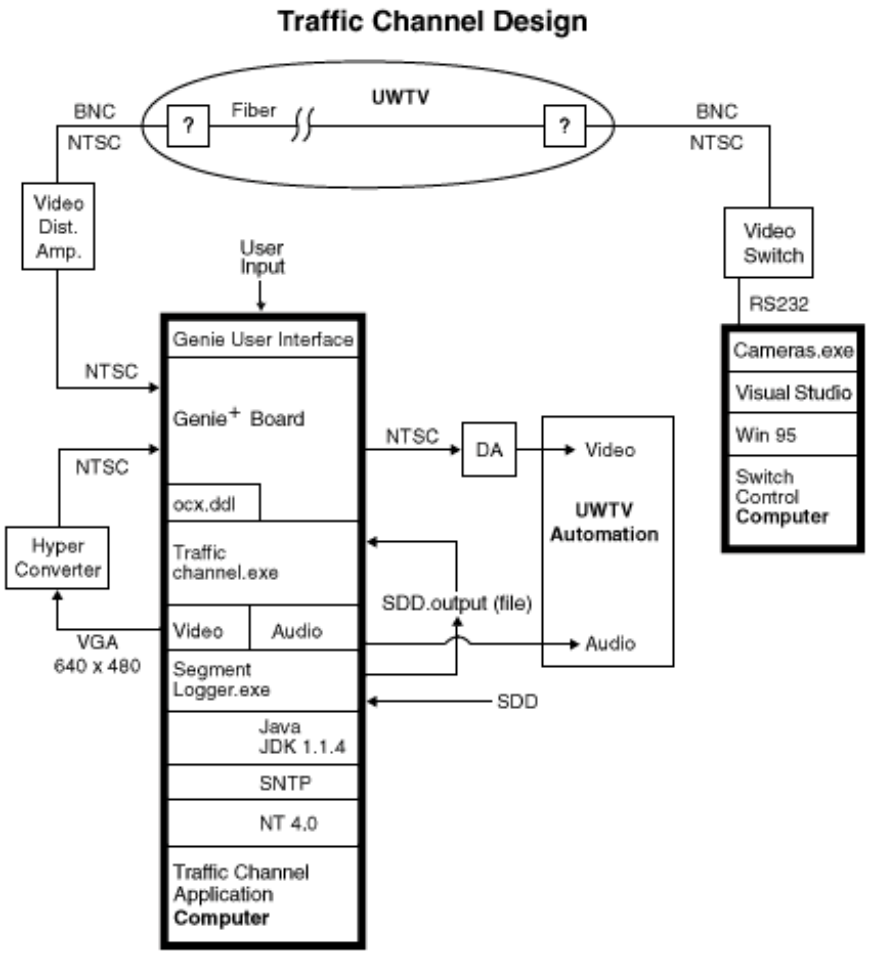


Figure 6-2. Traffic Channel design

6.2.1 The Display Computer

- An Intel PentiumII computer with at least 64MB RAM
- Windows NT version 4.0 and Service Pack 3
- VGA capable video adapter and monitor
- SoundBlaster compatible audio card
- IOmega zip optical drive (http://www.iomega.com/)
- A PC Video Conversion Hyperconverter
- An Internet connection with verifiable connectivity to the ITS backbone data
- An installed version of Java JDK, version 1.1.5
- A GeniePlus Digital Video Effects card (DVE). (http://www.pinnaclesys.com/)

6.2.2 The Video Control Computer

- An Intel compatible PC capable of running Windows 95
- A COM1 output serial device
- At least 32MB of system RAM
- An Internet connection for obtaining accurate system time

6.3 Installing the Traffic Channel from the Distribution Zip Disk

Operation of the Traffic Channel system requires the installation of two computer systems and three separate processes. The Display Computer must be physically located at the broadcast point and within connection of the desired video feed. This machine runs the **TrafficChannel.exe** map display program and the Java-based generic redistributor data acquisition program. The Video Control Computer must be located at the WSDOT computer room within physical proximity of the video control switch. This machine runs the **cameras.exe** camera control program.

6.3.1 Installing the Display Computer

1. Copy the **\TrafficChannel** directory from the distribution disk to the root of your C: partition.
2. Copy the **\segmentLogger** directory from the distribution disk to the root of your C: partition.
3. Verify that your computer is connected to the network.
4. Install the timekeeper software. Copy the **tardisnt3.zip** file from the Traffic Channel installation directory to a temporary directory on your machine. Unzip the the **tardisnt3.zip** file and install, following the directions provided with the software
5. Install the Java jdk version 1.1.5. Copy the **jdk115-win32.exe** file from the *\segmentLogger\JavaDistribution* directory to a temporary directory on your machine and run it to install Java.
6. Install the GeniePlus Digital Video Effects card following the instructions that are supplied with the hardware from Pinnacle Systems. (http://www.pinnaclesys.com/)

6.3.2 Installing the Video-Control Computer

1. Verify correct installation of Windows95 on your machine.
2. Verify that your computer is connected to the network.
3. Install the timekeeper software. Copy the **tardisnt3.zip** file from the distribution disk to a temporary directory on your machine. Unzip the the **tardisnt3.zip** file and install, following the directions provided with the software
4. Copy the *\TrafficChannel\VideoController* directory from the distribution disk to the root C: partition of the Video Control Computer
5. Connect the 9-pin, D-shell connector from the 9600 baud port of the video switch to COM1 of the Video Control Computer.
6. Run the **\VideoController\cameras.exe** program .

7. Connect a stand-alone video monitor to verify that the current video sequence (Section 6.3.2.1) is being affected.

Both the Display Computer and the Video-Control Computer rely on accurate time to synchronize the changing of video scenes with the changing of maps and for synchronization with a television broadcast system. The **tardisnt3** application above should be configured to use an available and accurate SNTP, (Simple Network Time Protocol), time server. Per the application's directions, it should be configured to keep the system clock within 100 msec of the referenced time base.

6.3.2.1 Current Traffic Video Sequence

A prototype demo of the proposed traffic channel application was created and posted on the web. A survey form gathered information about preferred camera views. Based on the results of this survey (see Figure 6-3), the following sequence of traffic videos is being sampled:

- State Route 520 Midspan
- Downtown - Yessler Way
- Interstate 90 Midspan
- Bellevue - NE 8th Street

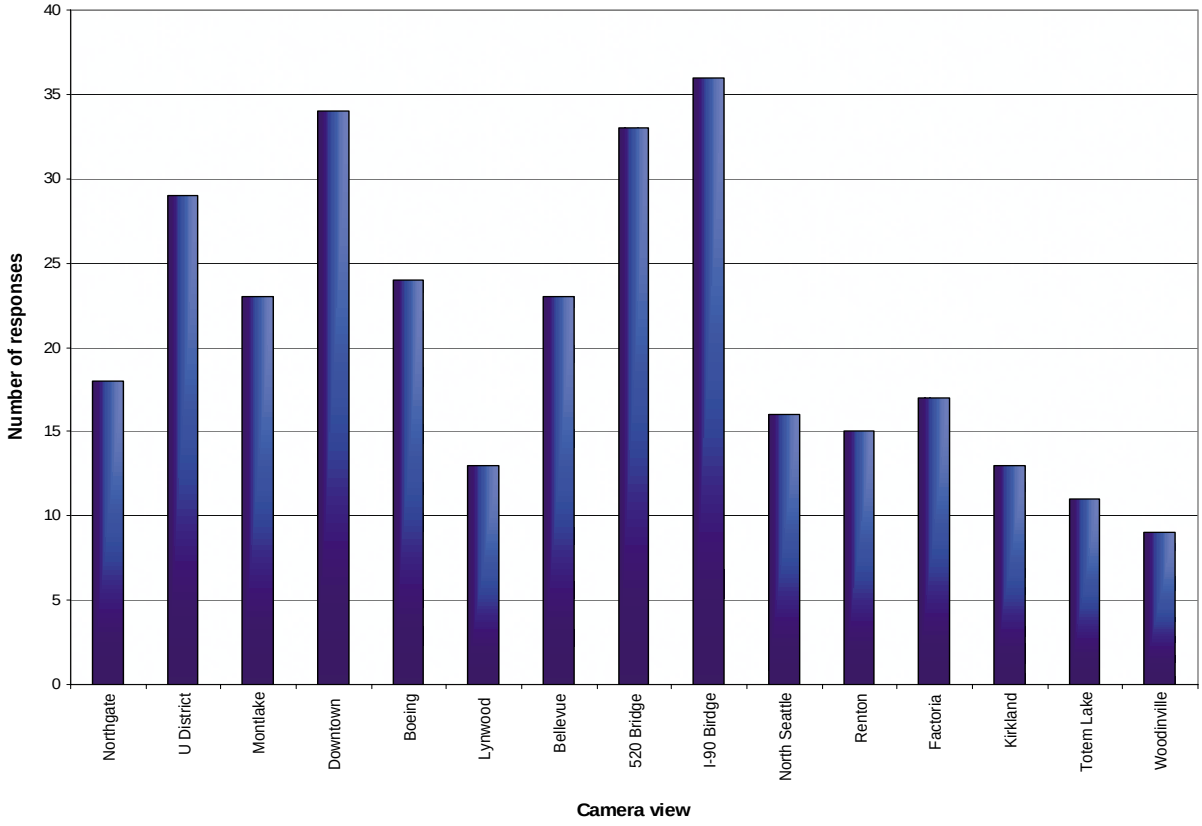


Figure 6-3. Camera views survey results

The sequence, which runs for 30 seconds, starts at the top of a minute or 30 seconds past the top of a minute in universal time. Each scene is selected for 7500 msec.

The camera selection mechanism for the scenes is built into the **\TrafficChannel\VideoController\switcher.cpp** program.

```
// The individual commands to switch the cameras
char *sequenceCmds[NCmds] = {
        "114#a",                // cctv509 520 midspan
        "22#a",                 // cctv105 Yesler way
        "49#a",                 // cctv859 I90 midspan
        "150#a"};               // cctv693 Bellevue NE
```

The mapping of camera names to switch port numbers can be obtained from the WSDOT CCTV database on the traffic control machine, harley (http://www.wsdot.wa.gov/regions/northwest/, Michael Forbis, (206) 440-4475).

6.4 Running the Traffic Channel Application

The **TrafficChannel.exe** application reads the real-time data gathered from the generic re-distributor and displays a sequence of maps based on the configuration files (Section 6.5).

The TrafficChannel application is run by starting the traffic data acquisition program and then starting the map presentation program.

To start the data logging application, open a DOS console window and enter the following commands:

```
cd \segmentLogger
run
```

To start the map presentation program, open another DOS console and enter the following commands:

```
cd \TrafficChannel
TrafficChannel
```

To start the **Cameras.exe** video control program, you must go to the WSDOT Traffic Systems Management Center in North Seattle (http://www.wsdot.wa.gov/regions/northwest/). The Video Control Computer is located in the WSDOT computer room there. Michael Forbis is the main contact for WSDOT technical assistance. He can be reached at (206) 440-4475. Then, open a DOS console on the Video Control Computer and enter the command:

```
\VideoControl\Cameras.exe
```

6.5 Configuring the Traffic Channel Map Display Application

The Traffic Channel application provides a large degree of flexibility through end-user modification of configuration files. This section uses examples from the UWTV implementation to present the end-user configurable components of Traffic Channel.

6.5.1 Defining a Video Sequence

The sequence configuration file defines a series of maps that that are presented in succession.

Note that the elements of the sequence configuration must be in the order presented. Single-line comments may be used with the # symbol.

```
# sequencer.txt
# The sequence definition for the UWTV TrafficChannel video presentation
#
# The UWTV presentation uses four maps, presented for 7500 milliseconds
# each for a total sequence time of thirty seconds.
#

4                # Number of maps in the sequence

# If map segments, defined in the following map file definitions
# have different foreground and background colors, then the specified
# colors are alternated this frequency
500              # Duration in msec of flashing map segments

# Sequence Sound File:
# This sound file is started at the beginning of the first map
# in the sequence and is repeated each time the first map is displayed.
Sounds\scrow.wav

# Time Synchronization Value:
# The start of the sequence is scheduled such that
# the sequence starts at the next second where the
# wall-clock time in seconds, modulus the sync value equals zero.
# The sum of the map display times in the following map definitions
# should equal this value.
30               # in seconds

# The Map Definitions:
# -map definition filename-    -duration of map display in msec-
TrafficMaps\mainlineWest.txt                7500
TrafficMaps\hovWest.txt                     7500
TrafficMaps\mainlineEast.txt                7500
TrafficMaps\hovEast.txt                     7500
```

6.5.2 Defining Display Maps

Map configurations are used to define the display appearance for an individual scene, or map, in the entire sequence (Section 6.5.1). The following sample illustrates and explains each of the components of a map display. It also shows how the components are defined in the configuration file. Note that the elements of the map configuration must be in the order presented. Single-line comments may be used with the # symbol.

This example is based on the fourth map in the UWTV implementation. In the Traffic Channel installation directory, itis the **TrafficMaps\hovEast.txt** file

```
# TrafficChannel map layout/configuration
# Definition to display HOV traffic data on the right side of
# the display and video on the right.

# number of video effects
2

# The Genie effect files and start times in msec
c:\TrafficChannel\Effects\pagetrn4.eff       5800  # transition between maps
c:\TrafficChannel\Effects\WestWipe.eff       1000  # swoosh out video on layer
2

# Map origin:
#   Origin is in the screen coordinate system, with a resolution
```

```
#   of 640x480 pixels, and is used
#   to adjust the relative position of the segment display and the
#   components that are displayed relative to the segments
244 0

#
# Color Table Definition
#    The color table must contain 16 entries numbered 0-15.  Each
#    color table entry is used to lookup a given object on the display.
#    When changing color definitions verify that the new selections
#    are 'NTSC safe' by previewing a sample on a vector scope
#    and waveform analyzer.
#
0  0     0   0    # Black;
1  255   0   0    # Flash Red;
2  0    24 127    # Dark Blue;
3  36   53 231    # Bright Blue, i.e. road sign bkgs
4  98  143 189    # Light Blue, i.e. lake background
5  0     0   0    # UNUSED
6  192 192 192    # Light Gray;
7  135 135 135    # Medium Gray;
8  75   75  75      # Dark Gray;
9  211  17  53      # Red;
10 0   210   0      # Green;
11 243 209  49      # Yellow;
12 255 116   4      # Orange
13 0     0   0      # UNUSED
14 0     0   0      # UNUSED
15 200 200 200      # White;

# segment closure information:
# -file name-           Name of file containing loop names of closed segments
# -closure color ix-    Index of color from above palette for closed segment
# -bad data ix-         Index of color for data not available
Closures\hovclosures.seg 0 7

# Segment Type Definitions
#  type#       : 0 - (N-1) where N is the number of segment types
#  capStyle    : horizontal | vertical
#  width       : width of the segment in pixels
#  borderWidth : width of the segment border in pixels
#  colorIndex  : 0-15 segment border color from the above color table
2   # number of segment types
0 horizontal 8 1 8
1 vertical  8 1 8

# Segment Definitions
265   # number of segments
SegmentDefinitions\hov.seg

# Static filled regions
# This section is used to draw the speed legend in the UWTV maps.
# The four (x,y) point pairs are connected with a single-pixel
# wide border and filled with the specified color.
#
6   # number of quadrilaterals to draw
# -x0- -y0- -x1- -y1- -x2- -y2- -x3- -y3- -color for fill- -color for flash-
#       If the foreground and background colors are equal, do no flashing
#       Color values must be indexes to the above color table.
70 330 80 330 80 350 70 350 9 6
```

```
70 350 80 350 80 370 70 370 9 9
70 370 80 370 80 390 70 390 11 11
70 390 80 390 80 410 70 410 10 10
70 410 80 410 80 430 70 430 0 0
70 430 80 430 80 450 70 450 7 7

# Decoration Declarations
#    Decorations are pre-defined .PNG image files to place on the display
35 # Number of decorations

# -image filename-  upper-X upper-Y  (position is relative to the map origin)
StaticImages\BackgroundEast.png -244 0
StaticImages\HOVBanner.png  -141 35
StaticImages\I405.png 235 105
StaticImages\I5.png 155 135
StaticImages\520.png 185 240
StaticImages\I90.png 195 332
StaticImages\Seattle.png 55 325
StaticImages\Boeing.png 55 370
StaticImages\Renton.png 250 430
StaticImages\Factoria.png 280 360
StaticImages\Bellevue.png 280 288
StaticImages\Kirkland.png 280 230
StaticImages\TotemLake.png 280 180
StaticImages\Woodinville.png 270 130
StaticImages\Plane.png 85 410
StaticImages\BellevueBanner.png -195 295
StaticImages\Closed.gif -160 412
StaticImages\NoData.gif -160 432
StaticImages\digit0.png -160 332   # build the legend strings, 0 - 19
StaticImages\hyphen.png -150 332
StaticImages\digit1.png -140 332
StaticImages\digit9.png -130 332
StaticImages\digit2.png -160 352   # 20 - 34 mph
StaticImages\digit0.png -150 352
StaticImages\hyphen.png -140 352
StaticImages\digit3.png -130 352
StaticImages\digit4.png -120 352
StaticImages\digit3.png -160 372   # 35 - 49
StaticImages\digit5.png -150 372
StaticImages\hyphen.png -140 372
StaticImages\digit4.png -130 372
StaticImages\digit9.png -120 372
StaticImages\digit5.png -160 392   # 50 +
StaticImages\digit0.png -150 392
StaticImages\plus.png -140 392

#
# ITS data file created by the Generic redistributor
c:\segmentLogger\sdd.output

# Data segments are alternated between the data dependent
# color and the static color to affect flashing.
# The data dependent values are determined in the followin Speed Legend
# -color index of data independent color-
6

# Speed Legend:
# number of speed quanta for dividing colors
# speeds are in triplets:
```

```
# -upperbound data value- -color index- -is flashable-
4        #  the number of colors to use
20 9    1
34 9    0
49 11   0
50 10   0  #  the top-end speed doesn't really need a lower bound

# Camera usage definition
# -Camera ID-     A string to identify which video camera to use
#                 for this map.  This entry is not used in the UWTV
#                 implementation for which the camera switching
#                 is controlled by the video control computer at WSDOT.
# -Segment Name-  Not Used
# -old img-       One of the map decorations above which is to be replaced
# -new img-       A decoration to use instead of -new img-
#
Bellevue Dont_Hilite_Segment StaticImages\Bellevue.png
StaticIMages\BellevueVideo.png
```

6.5.3 Defining Segments

Segment definitions are constructed from three components. The first is two pairs of coordinates arranged as x1, y1, x2, y2 in the screen pixel coordinate system. This specifies the endpoints of a road segment. (Note that the specification of segment width, segment border width, and the coupling of color to loops data value is specified in the Segment Type Definitions section of the map configuration file.)

The second component of each entry is a cap style value. This indicates whether the connecting lines at each end of the segment are to be drawn horizontally or vertically. If the style is "1," the end lines are drawn vertically; otherwise, the end lines are drawn horizontally.

The last component of each entry is the 15-character loop identifier that maps the roadway sensor data onto the segment. Roadway data is obtained from the generic redistributor (the real-time data gathered in the **\TrafficChannel\segmentLogger\sdd.output**). The value of the data is used to select the color for the segment.

Single line comments can be entered in the segment definition files using the # symbol.

Segment definition files are stored in the *\TrafficChannel\SegmentDefinitions* directory. The following sample is a partial listing from the **PugetSound.seg** file, which is used for both of the "Mainline" traffic displays.

```
148 267 157 267 1 ES-502D:_MW_Stn  # 520 west
157 267 166 267 1 ES-504R:_MW_Stn  #
166 267 175 267 1 ES-506R:_MW_Stn  #
175 267 184 267 1 ES-506R:_MW_Stn  #
184 267 193 267 1 ES-506R:_MW_Stn  #
193 267 202 267 1 ES-506R:_MW_Stn  #
202 267 211 267 1 ES-514D:_MW_Stn  #
211 267 220 267 1 ES-516R:MMW_Stn  #
220 267 229 267 1 ES-519R:_MW_Stn  #
229 267 238 267 1 ES-520D:_MW_Stn  #
238 267 247 267 1 ES-521R:MMW_Stn  #
247 267 256 267 1 ES-524R:MMW_Stn  #
256 267 265 267 1 ES-528D:_MW_Stn  #
265 267 274 267 1 ES-531R:MMW_Stn  #
274 267 283 267 1 ES-533D:_MW_Stn  #
283 267 292 267 1 ES-535D:_MW_Stn  #
292 267 301 267 1 ES-538D:_MW_Stn  #
301 267 310 267 1 ES-541D:_MW_Stn  #
148 280 157 280 1 ES-502D:_ME_Stn  # 520 east
```

```
157 280 166 280 1 ES-504R:MME_Stn  #
166 280 175 280 1 ES-506R:MME_Stn  #
175 280 184 280 1 ES-506R:MME_Stn  #
184 280 193 280 1 ES-506R:MME_Stn  #
193 280 202 280 1 ES-506R:MME_Stn  #
202 280 211 280 1 ES-514D:_ME_Stn  #
211 280 220 280 1 ES-516R:_ME_Stn  #
220 280 229 280 1 ES-519R:MME_Stn  #
229 280 238 280 1 ES-520D:_ME_Stn  #
238 280 247 280 1 ES-521R:_ME_Stn  #
247 280 256 280 1 ES-525R:MME_Stn  #
256 280 265 280 1 ES-528D:_ME_Stn  #
265 280 274 280 1 ES-531R:_ME_Stn  #
274 280 283 280 1 ES-533D:_ME_Stn  #
283 280 292 280 1 ES-535D:_ME_Stn  #
292 280 301 280 1 ES-539D:_ME_Stn  #
301 280 310 280 1 ES-541D:_ME_Stn  #
```

6.5.4 Segment Closure Files

Segment closure files are used to select roadway segments (previously defined in the segment definition file, Section 6.5.3) which are closed or which for other reasons should be drawn in the color specified for "No Data Available." These files are typically used for the HOV display (Section 6.1, images 1 and 3) where a large number of segments do not have HOV lanes but are needed to visually complete the map.

Each line in the closures file must contain one string which is the identifier of a roadway segment that was defined for the map in which this file is used.

Closure files are stored in the *\TrafficChannel\closures* directory. The following sample is a partial listing of the **hovclosures.seg** file:

```
ES-502D:_MW_Stn
ES-504R:_MW_Stn
ES-506R:_MW_Stn
ES-506R:_MW_Stn
ES-506R:_MW_Stn
ES-506R:_MW_Stn
ES-514D:_MW_Stn
ES-516R:MMW_Stn
ES-519R:_MW_Stn
ES-520D:_MW_Stn
ES-521R:MMW_Stn
ES-524R:MMW_Stn
ES-528D:_MW_Stn
ES-531R:MMW_Stn
ES-908R:MMW_Stn
...
```

6.5.5 Genie Video Effect Files

Genie video effects files are binary files created and saved with the GeniePlus software switcher program. See the printed and online documentation provided with the Genie DVE hardware. (http://www.pinnaclesys.com/)

Video effect files used with Traffic Channel are located in the directory *\TrafficChannel\Effects*. These files may be previewed and modified within the GeniePlus program. (http://www.pinnaclesys.com/)

6.5.6 Static Image Files

Static image files are used in Traffic Channel to present those images on the display map which do not change, e.g. roadway banners, town name banners, the map background, and video banners. Image files must be in Portable Network Graphics format (.PNG) and must be no larger than 640x480 (the area of the map background). All colors used in the image file (Section 6.5.6.1) must be defined in the 16-color palette of the Color Table Definition of the map configuration file (Section 6.5.2). Image files are stored in the directory *\TrafficChannel\StaticImages*.

Images are located on the map by specifying an (x,y) pixel coordinate which is used to place the upper-left corner of the image. Note that images must fit within the 640x480 pixel region of the display.

Images are either drawn on the screen or rendered in the order presented in the map configuration file (Section 6.5.2). If image locations overlap, the images drawn later will cover, or sit on top of, those drawn earlier. Therefore, images such as the background should be listed first in the configuration file.

The Adobe Photoshop program (http://www.adobe.com/) is recommended for previewing, modifying, and creating PNG files for use with Traffic Channel.

6.5.6.1 Specifying Colors for Use on TV

Computer-generated colors often differ in appearance when viewed on a television video monitor rather than on a computer video monitor. The analog, NTSC, signal that is used in televisions has limitations and unique characteristics. Colors generated on the computer need to be carefully previewed, reviewed, and electronically tested before use on a television video monitor.

In the **TrafficChannel.exe** program, as in most computer implementations, color is specified by quantitatively mixing the colors red, green, and blue. That is an RGB color system. The desired amount of each color is specified as a quantity from 0 to 255. For example, pure red would be specified as 255 0.

In general, the range of possible colors is smaller for the television than for the computer. When the computer generates a color that is outside the television's display range capabilities, there will be artifacts such as aliasing, color wash out, and elements shadowing displayed on the television.

The following "Rules of Thumb" can be used for specifying safe colors:

* Avoid specifying pure colors; that is, colors that have a non-zero value for only one of the RGB quantities.
* Avoid selecting any R, G, or B value that is within 20% of maximum; that is, 204 (255 - 20%).
* Use a program such as the Adobe Photoshop (http://www.adobe.com/) NTSC Safe tool to test your selections.
* Enlist the assistance of a video professional to preview your selections on a vector scope and waveform monitor. Most video professionals, such as the staff at UWTV, are experienced with these instruments and can immediately verify your selection. Al Ross is the main contact for technical assistance at UWTV (http://www.washington.edu/uwtv/), which is located on the University of Washington campus in Kane Hall, UWTV broadcast offices. He may be reached at (206) 543-9926.

The above list is a conservative starting point. There can be exceptions, but always verify your selection with a video monitoring instrument.

The Traffic Channel application supports up to 16 colors as specified in the Color Table Definition section of the map configuration file (Section 6.5.2). All elements on the display - banners, static regions, segments, and image file decorations - must contain only colors available in this palette.

6.5.7 Sound Files

The sequence configuration file (Section 6.5.1) references a single sound file which is played at the start of the first map displayed in the sequence. The sound file is started at the beginning of the sequence and is restarted each time the sequence repeats. Sound files should not be longer than the sum of all the sequences. Sound files must be in Windows .WAV format.

6.6 Troubleshooting Traffic Channel

When correctly configured, the Traffic Channel application is designed to run autonomously without intervention. However, because of the numerous hardware assets that the program relies on, failures can occur. Correct operation requires that all components illustrated in this document be operating correctly.

The **TrafficChannel.exe** program writes diagnostic information to the **\TrafficChannel\ErrorLog.txt** file as it runs. This file should be periodically reviewed and should be the first place to look when the program appears to be running incorrectly. Note that this file is appended to each time the program is run, so it should periodically be purged.

Common failure symptoms and possible diagnoses:

*I start the **TrafficChannel.exe** program but it does not appear on the computer.*

- Note that the program *sleeps* or waits to start until the synchronization value specified in the sequence configuration file. Verify that this value is appropriate and that the time you are waiting is more than the synchronization value.
- The Genie DVE board is memory-usage intensive. No other desktop applications should be running when **TrafficChanel.exe** is running. If you have exhausted memory, the system should be rebooted.

*I start the **TrafficChannel.exe** program and it appears on the computer monitor but not the video display.*

- Re-seat the physical cable connections to the pigtail on the DVE board and the video monitor.
- Reset the power on the video monitor.
- Reset the power on the DVE board; that is, shutdown, turn off the power on the computer, and reboot.

The video displayed in the DVE swoosh image is not the correct image for a given map.

- Verify that the video-control computer is operating correctly.

The video signal displayed in the swoosh of the DVE appears black or as white noise.

- Disconnect the traffic video signal cable from the DVE board and connect directly to a video monitor to verify the signal. If there is no signal coming out of the fiber, contact traffic personnel at WSDOT (http://www.wsdot.wa.gov/regions/northwest/, Michael Forbis, (206) 440-4475) to see if there is a known failure in the video switch or acqusition equipment.
- If the video previews correctly, reset the DVE board.

The program is running but the segment colors are not changing.

- Verify that your network connection is up by using ping or other network software diagnostics. Verify that you can reach the ITS generic re-distributor machine by pinging the machine *nova.its.washington.edu*.

- Verify that the segmentLogger utility is running and that the file it creates, **\segmentLogger\sdd.output**, is being updated.

*Error or warning dialogs are displayed when starting the **TrafficChannel.exe** program.*

- One or more of the configuration files have been corrupted or incorrectly modified.
- System type error messages typically indicate an OS-related failure that has led to the exhaustion of resources. Time to re-boot.

6.7 Tools and Tips for Modifying Traffic Channel

In addition to the system requirements, the following development tools are required to modify Traffic Channel components:

- Microsoft VisualStudio with Microsoft C++, version 5.0 (http://www.microsoft.com/)
- Adobe Photoshop (http://www.adobe.com/)
- PaintShop Pro (http://www.jasc.com/)

All executable components for Traffic Channel are built using Microsoft Visusal Studio Workspace files (.DSW). Workspace files provide cohesive access to all files in a given project and provide *makefile* dependancy checking. To open a specific workspace file, select “File | Open Workspace” from the VisualStudio menu bar. To open the project for the **TrafficChannel.exe** program, select **\TrafficChannel\SourceCode\TV**. The project for **Cameras.exe** is located at **\TrafficChannel\VideoController\Cameras**.

7. UW Cable Television "Traffic TV" Evaluation [9]

7.1 Project Description

Traffic TV is a traffic information program broadcast to subscribers of AT&T Cable Services[10] in the Seattle region at various times during the morning and evening commute periods and over the noon hour. The program format and presentation were designed and produced by the University of Washington, and it is broadcast on local cable TV Channel 76, one of two educational channels produced by the University of Washington. Traffic TV has been placed on the air for a limited period of time to assess its value to Seattle area travelers. It is sponsored and financed under the Smart Trek MMDI program.

The Traffic TV program was aired initially on June 1, 1998, and reached an estimated 60,000 households. AT&T Cable Services has been building out its service in the region and increasing the number of households that can access this channel. By early 1999, AT&T estimated that about 105,000 households had viewing access to Traffic TV, primarily in King County. AT&T anticipated completing its digital build-out by March 2000, at which time Traffic TV was expected to be available in 484,000 households in King and Pierce Counties. Figure 7-1 presents two typical displays for Traffic TV.

The Traffic TV broadcasts are shown a total of 10 and a half hours a day, from 5:00 AM to 9:00 AM, 11:00 AM to 1:30 PM, and 3:30 PM to 7:30 PM, seven days a week. They are composed of a 30-second sequence that displays four Seattle region traffic speed maps (taken from the Washington State DOT traffic Web site) for a period of 7.5 seconds each. The broadcast rotates 15 minutes of Traffic TV, followed by 5 minutes of Seattle region satellite weather maps. Traffic video images acquired from four different roadside cameras display live video of the major congestion points in Seattle and Bellevue. The map images alternate between main line traffic and high occupancy vehicle lanes. The live video images alternate from Seattle, Bellevue, and both the bridges over Lake Washington. Color-coded map displays show traffic speeds. Background music is played during the broadcast, and there are no spoken messages.

The MMDI costs to deploy Traffic TV are presented in Table 7-1 below. The major costs of this project were the labor costs for the development of the system. Of note, annual operations and maintenance costs for the project are estimated to be approximately 38 percent of the deployment costs. The relatively high costs of operations are mainly due to the annual television airtime charges for the program.

7.2 Evaluation Description

Traffic TV was selected as one of the ATIS projects under Seattle's Smart Trek program to be evaluated from a customer satisfaction standpoint. The objectives of the Customer Satisfaction evaluation of the Traffic TV were to assess the extent of awareness of this service:

- Understand who was using this service, including when and how they use it, and describe the characteristics of users and non-users.
- Understand what aspects of the service are considered more useful or beneficial than other aspects, and what kinds of improvements or modifications are suggested that could enhance user acceptance and satisfaction.
- Understand whether and how use of Traffic TV affects driving decisions and behaviors, including the factors that help explain these behaviors.

---

[9] Excerpted from Metropolitan Model Deployment Initiative Seattle Evaluation Report Final Draft, Publication No.: FHWA-OP-00-020, May 30, 2000, U.S. Department of Transportation, ITS Joint Program Office, HVH-1, 400 7th Street SW, Washington, DC 20590, Phone: (202) 366-0722, Facsimile: (202) 493-2027

[10] Formerly TCI Cable, Inc.

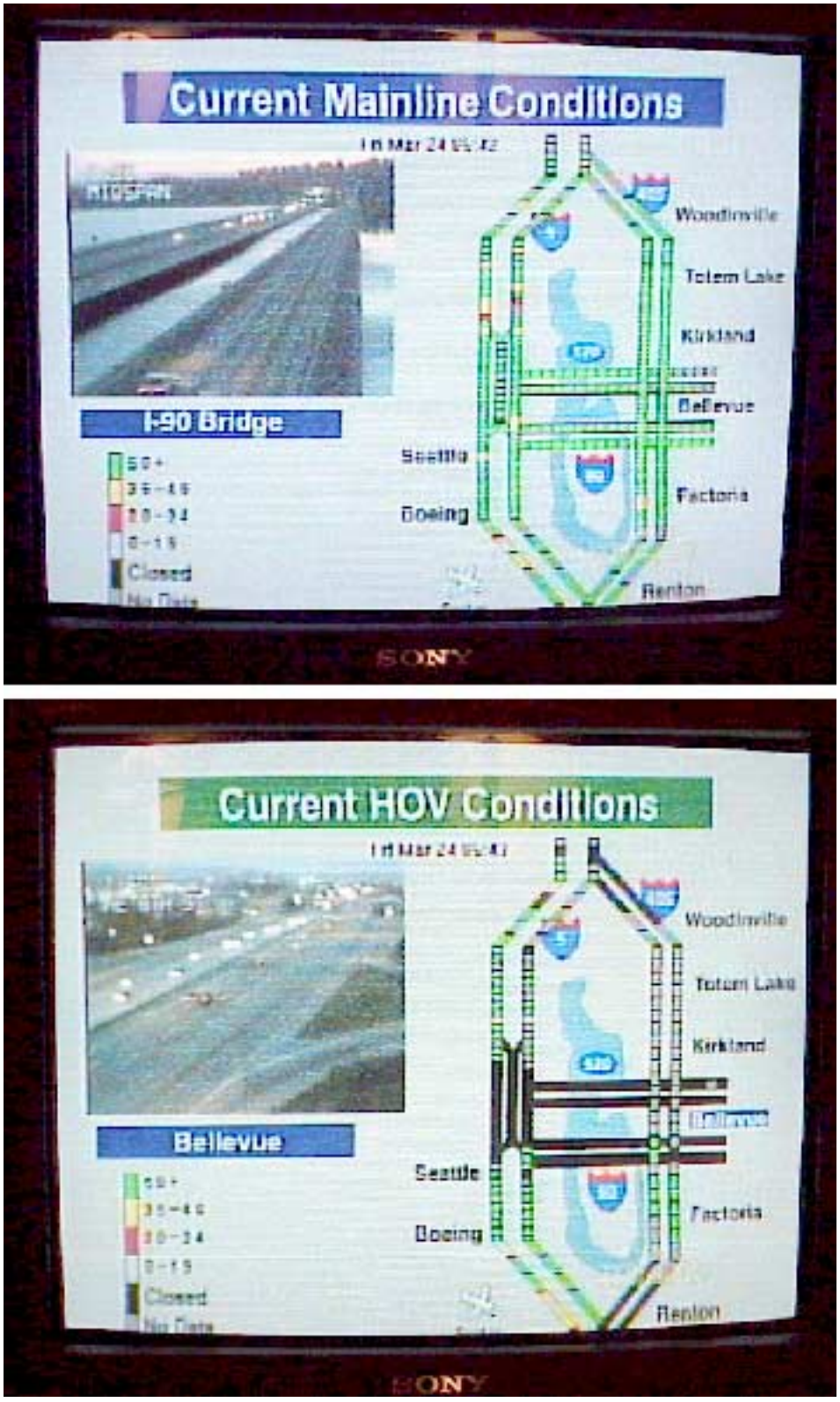


Figure 7-1. Traffic TV typical displays

Table 7-1. Cable Television Traffic TV Cost Etimate [11]

Equipment Description		Non-Recurring Costs		Recurring Costs
Hyperconverter	$	3,484		
Pentium Pro 200 Mhz	$	4,767		
Video Converter Card, DeMux and Mux	$	8,265		
Other Equipment (Proposed Budget)	$	40,000		
Supplies & Materials	$	2,685		
Labor (Including Indirect Costs & Benefits)	$	170,805		
Contractual Services	$	3,700		
UWTV Station Airtime Charge	$		$	78,000
14% Share of 3 Pentium Workstations & Associated Equipment	$	3,393		
14% Share of Labor (including indirect costs & benefits)	$	78,324		
14% Share of Other Direct Costs		3,585		
14% Share of Hardware & Supplies (replaced every 2 years)			$	1,696
14% Share of Fiber Link & Other Contractual Services			$	2,428
14% Share of Operations Labor (3 UW FTEs)			$	40,178
Total	**$**	**319,008**	**$**	**122,302**

- Assess the value and benefits of Traffic TV to users.
- Provide recommendations for enhancing the benefits of real-time traveler information provided over cable TV.

The evaluation had two components: a qualitative focus group and a quantitative mail-out survey questionnaire. The purpose of focus groups was to develop in-depth understanding of attitudes and behavior, and to help shape the questions for the survey. The remainder of the discussion below focuses on the mail-in survey administration and results.

Working closely with AT&T Cable Services, a stratified, random sample of 10,000 households with access to this program was identified from 21 Seattle zip codes (all on the west side of Lake Washington). Somewhat less than half of all households in the sampled zip code areas were believed to have access to Traffic TV, though there was a lot of variability across zip code areas due to the different stages of AT&T's service buildout. Questionnaires were mailed to the households during the week of April 19, 1999, under Smart Trek letterhead. For legal reasons, no incentives were offered to encourage responses. As of June 1, 1999, 1,705 (17 percent) completed questionnaires had been returned (226 viewers and 1,479 non-viewers). This sample of viewers constitutes 13.3 percent of the returned questionnaires, which is an estimate of the viewership rate in the population. Of the 1,479 non-viewers, 226 were randomly selected and included in the analysis. The 452 returns were edited, cleaned, keyed, and verified and made available for analysis July 29, 1999. After cleaning the data, the final sample included 223 viewers and 222 non-viewers for a total working sample of 445.

In order to allow the survey findings to represent the population of households with viewing access, the sample was weighted to adjust for differential non-response by zip code and to scale up the non-viewer component of the sample to reflect the actual number of returns from non-viewers by zip code.

[11] Note that a 14% share of the ITS Information Backbone System cost elements have been included here as part of this cost estimate since relevant ITS information is either currently exchanged between this system and the Backbone, or because this system has the potential to interface with the Backbone in the near future.

7.3 Results

7.3.1 Respondent Characteristics

Respondents in the weighted, representative sample are older, better educated, and have higher household incomes compared with the general metropolitan area population. Most also report that they personally use at least once a week a variety of high technology communication and information devices, such as personal computers at home (64.0 percent) or at work (62.8 percent), the Internet at home (51.2 percent) and at work (45.7 percent), and cell phones (50.5 percent). The majority commute at least three days in an average week (75.1 percent), mostly in personal vehicles (84.2 percent). Public transportation other than a ferry is used by 15.9 percent. The sample is split between those who say they regularly watch television weekday mornings or evenings (49.8 percent say they do watch TV). As expected, respondents who say they have ever watched Traffic TV are more likely to regularly watch morning or evening television, but 37.5 percent of Traffic TV viewers say they don't regularly watch television.

7.3.2 Awareness of Traffic TV

Viewers of Traffic TV were asked to indicate how they first found out about this traffic program. Most reported that they found it while flipping through the channels on their television (85.8 percent). A few found out from friends (3.9 percent), and some learned about it by having received the survey (2.4 percent). The remainder reported other ways or said they couldn't remember how they first heard about it. This helps explain why more people whose households are connected to cable with access to this program have never seen the program. The higher numbered broadcast channels may not readily be encountered through casual channel surfing, and there has been no formal campaign to advertise or promote this traffic information service.[12]

7.3.3 Who used Traffic TV, when/how they used it, and user/non-user characteristics

While the Traffic TV program has been available for viewing by these respondents for about a year, the sample is composed mostly of very recent viewers. Among viewers, 43.3 percent had been watching for one month or less, just over half (50.6 percent) had watched for two months or less, and 88.0 percent had watched for six months or less prior to the survey. More reported viewing Traffic TV in the mornings (41.1 percent) than the afternoons (25.0 percent) or during mid-day (11.1 percent). This is logical, given that 78.4 percent of the users of Traffic TV are commuters, and commuters are less likely to have access to a TV at their office during the day. Commuters are as likely as non-commuters to view Traffic TV in the mornings (41.2 percent vs. 40.0 percent), but much less likely than non-commuters to view mid-day (9.1 percent vs. 18.4 percent) or afternoons (21.7 percent vs. 37.5 percent).

Users are more likely to be male (62.0 percent male vs. 38.0 percent female). Education and income appear unrelated to having ever viewed Traffic TV. Traffic TV viewers in Seattle were less likely to be elderly, in contrast to traffic TV viewers in Tempe, AZ, who were more likely to be elderly.

7.3.4 Frequency of Use

A little less than one-third of the viewers of Traffic TV (29.2 percent) watch the broadcasts more than once a week, 31.3 percent watch two to four times a month, and 39.5 percent watch less than once a month.[13] Those who have been viewers for the longest period of time are much more likely to be frequent viewers than those

---

[12] Note that WSDOT did issue a press release announcing this service, and several of the news media, including the local network TV stations, ran stories on Traffic TV.

[13] After adjusting for non-responses and invalid responses to the frequency of use question in the weighted sample, 57 respondents report that they view TrafficTV more than once a week. These are our "frequent viewers," who constitute approximately 4 percent in the population.

who recently started watching. Over half (56.7 percent) of the viewers have been viewers for two or more months, and 40.4 percent of them watch more than once a week.

7.3.5 Aspects of Traffic TV considered most useful or beneficial, and suggested improvements or modifications to enhance user acceptance and satisfaction

The survey asked respondents who had seen Traffic TV to agree or disagree with opinion statements describing different aspects of program, such as how information is presented, the content of the program, and its value to the viewer. Overall viewers responded as follows:

- 75.4 percent would like to be given more information about the type and extent of incidents, special events, and trouble spots.
- 65.4 percent want an indication of the direction of traffic flow as shown on the camera view.
- 63.9 percent would like the same type of information to always appear on the same part of the screen;
- 63.8 percent would like the broadcasts to suggest alternative route possibilities when there are conditions that slow or block traffic.
- 60.5 percent of the respondents would like to hear a voice describing what is happening on the maps.
- 54.7 percent feel that the explanation of the different traffic speeds is too small to read easily.
- 53.2 percent would like the broadcast to help them decide whether road conditions make it unsafe to drive.
- 45.0 percent feel that the way in which the screens change from one view or map to another is distracting.
- 26.0 percent would like to have a number they could phone with suggestions for improvements to Traffic TV.

In addition, several statements ask about the usefulness and accuracy of Traffic TV:

- 44.6 percent agree that the broadcasts provide adequate coverage about travel conditions along the routes they travel.
- Only 5.0 percent find that the information on Traffic TV is inaccurate.
- 43.1 percent find the weather information on Traffic TV useful.
- 47.9 percent agree that Traffic TV helps them to avoid traffic congestion.
- 34.2 percent report that Traffic TV lets them estimate how long their trip will take.

Those who had ever viewed Traffic TV were also asked if they had any other comments about how Traffic TV could be improved to make it more useful. Far and away the most commonly suggested improvement was for more cameras and camera coverage, especially to provide coverage of arterials and major intersections. Some respondents felt that Traffic TV should be better publicized. A few viewers suggested showing maps by districts and collecting and displaying additional information. In addition, several mentioned specific areas that they would like to see added to Traffic TV.

7.3.6 Value and benefits of Traffic TV to users

Frequent users place significantly greater value on Traffic TV than do less frequent viewers. While, overall, only 6.9 percent of the viewers report that they would be willing to pay an extra $1 each month with their cable bill

for Traffic TV, frequent viewers are much more likely than occasional viewers to report that they would be willing to pay (16.1 percent versus 0.8 percent).

7.4 Summary and Discussion of Major Findings

The Major Findings of this analysis are summarized and discussed below:

- Out of the 223 respondents who said they had ever watched the Traffic TV broadcasts, 85 percent said they found out about the broadcast while flipping through the channels on their TV. This is similar to what we learned about the traffic TV program in Tempe, Arizona, and indicates a fairly low level of awareness and use of this program in the population. It also suggests the potential value of promotional efforts to let cable viewers know about this program. Respondents even suggested more attention be given to promoting the program. Viewers of Traffic TV are much more likely to report that they regularly watch television weekday mornings or evenings (63.2 percent) than those who say they have not viewed the Traffic TV broadcasts (45.2 percent), and this is consistent with how they stumbled upon the Traffic TV program.

- Viewers of Traffic TV report that they also use a variety of other technologies, such as personal computers, the Internet, and pagers and are more likely to use these compared with non-viewers. However, when we look at the subset of viewers who are the most frequent users of Traffic TV, we find they are less likely to use these other technologies, though the differences generally are not statistically significant. These frequent users of television for traffic information fit the segment profile called "low-tech pre-trip info seekers" as defined and analyzed in a companion report.[12]

- Among all the users of Traffic TV, 45 percent say they use the broadcasts for commuting, and among just the regular commuters (who represent 75 percent of our entire sample), 55 percent use Traffic TV for commuting. The non-commuters are much more likely than commuters to use the program for other purposes, such as visiting friends, shopping, and recreation. Frequent viewers of the broadcasts also say they use the program for a wider variety of trip purposes compared with the infrequent viewers.

- Most commuters (80 percent) experience congestion that, on average, lengthens their normal trip by 8.6 minutes over what it would be if traffic were free-flowing. Most commuters (83 percent) disagree (mild to strong disagreement) with the statement: "I rarely encounter unexpected traffic congestion," and almost half (48 percent) agree with the statement: "At least twice a week there's an unexpected delay on my route." The frequent viewers of Traffic TV are much more likely to experience congestion than less frequent viewers and non-viewers. On average, congestion adds 13.2 minutes to their commute every day, unless an unexpected event occurs to further increase their commute time.

- Route changes were the most likely choice for viewers who commute and said they had consulted any source of traffic information. The most frequently indicated behaviors are route changes (31 percent take a mostly different route; 22 percent make small route changes) and trip timing changes (20 percent leave earlier and 14 percent leave later). The reasons for these changes are to avoid congestion (94 percent say this is important to them), saving time (91 percent), using time more effectively (80 percent), reducing stress (74 percent), reducing the risk of an accident (40 percent), saving gasoline (32 percent), and saving miles (18 percent).

- Severe weather occurs from time to time in the Seattle region, and we asked commuters how they respond when learning from Traffic TV broadcasts about weather-related problems on their route. Under these conditions, such as high wind, heavy rain, snow, and ice, leaving earlier or postponing the trip altogether are much more likely responses than under normal congestion.

- In this survey we asked viewers of Traffic TV how they would respond when they learned that their trip from home to work/school would take 15 minutes longer than normal. Then we asked the same question under the condition of a 30-minute delay. Both these times exceed their average congestion delays by large amounts. Leaving earlier is the most frequently selected response given a 15-minute delay,

followed by small route changes and large route changes. But when the delay is doubled to 30 minutes, respondents select each of these three options more often, and are much more likely to select a large route change. Among those who say they are less likely to make the small route change when faced with the 30-minute delay, 74 percent of them increase the frequency of selecting a large route change.

Backbone

8. A Self Describing Data Transfer Methodology for ITS Applications [14]

8.1 Introduction

The SmartTrek, Model Deployment Initiative, ITS Backbone is a set of protocols and paradigms designed to tie multi-agency geographically distributed ITS applications together. Applications, in this context, are a set of components that: extract ITS data from multiple source agencies, modify that data to create derived information products, distribute the data to Independent Service Providers (ISP's), warehouse the data to create data mines, and use the data to create traveler information. The backbone is built around two key concepts: 1) a Component Architecture, described in "A Structured Approach to Developing Real-Time, Distributed Network Applications for ITS Deployment" [1] and 2) Self Describing Data (SDD) support, the topic of this paper.

This paper first presents an overview of the structure of SDD and discusses relevant standards. Two related standards, the National Transportation Communications for ITS Protocol (NTCIP) [2] and the Transportation Network Profile of the Spatial Data Transfer Standard (SDTS) [3], are used to provide a context for SDD. This paper describes the methodology used to create SDD and presents an example using traffic management data. It further details the actual implementation and use of SDD in the Smart Trek Model Deployment and describes an SDD receiver used to support several applications in both traffic management (North Seattle ATMS) and traveler information (TrafficTV, BusView, Transit Watch) applications. Finally, this paper describes the publicly available JAVA toolkit that implements SDD receivers and demonstration applications. The demonstration applications allow anyone with an Internet connection to have both traffic and transit data from the Seattle area as well as providing as an automated database builder.

This paper provides a framework for an enabling technology to share traffic and traveler information between public agencies as well as providing a clearly defined open interface between public agencies and the private sector.

8.2 Background

The wide variety of remote sensors used in Intelligent Transportation Systems (ITS) applications (loops, probe vehicles, radar, cameras, etc.) has created a need for general methods by which data can be shared among agencies and users who own disparate computer systems. Such data sharing requires that the sender and recipient of the data agree on a method for transfer. To date, most systems constructed for this purpose either lack generality or are limited to data transfers of a specific type [4, 5, 6] or they are so general and complex as to be very difficult to implement [3]. The work presented in this paper is aimed at creating a general mechanism for *self-describing* data transfers of data streams that are produced by a set of remote sensors that change in number and type as a function of time. We present our self-describing data transfer concept in the context of ITS applications; however, our approach is applicable to a variety of data types and sensors. Self-describing data transfer requires information about the meaning of the data to be included as part of the transfer [7]. This *meta-data* must include all information needed to interpret the actual data stream. For example, the time-invariant properties of a remote sensor that might be relevant include its location, the units of measurement, the precision of its measurements. In addition, a description of the algorithm used to extract the desired information from the data is required. Any successful methodology that provides self-describing data transfers must meet the following criteria:

1. The transfer includes all information (*meta-data*) needed to interpret the data together with the data itself. If this requirement is met, the data transfer is *self-describing*.

---

[14] Paper presented at the 78th Annual Transportation Research Board Meeting in Washington, D.C., January 10-14, 1999. Authors: D.J. Dailey, D. Meyers, N. Friedland.

2. The transfer method can be applied to a broad category of data types and procurement methods. This is a requirement for *data type independence* of the data transfer method.

3. The transfer method is applicable to a wide variety of computing environments. This is a requirement for p*ortability and general applicability* of the data transfer method.

Strictly, only the first requirement need be met to qualify the data transfer as self-describing. The effect of the other requirements is to enhance the generality of a transfer method. There are a variety of proposed data transfer mechanisms that transfer the data and meaning [8, 9, 10, 5, 6]. Many of the data sharing methods used to date have involved the construction of custom software that "understands" the meaning of the data to be transferred for each class of data transfer [6, 5]. Such methods fail the second and third criteria listed above. They fail the second criterion because the transfer is specific to one type of data. They fail the third criterion unless the custom software is written in a very portable manner.

We present a new approach to solving the self-describing data transfer problem. Our data transfer method serializes a data description in the form of a data dictionary with the actual data to be transferred. Our data description makes use of the power of database query languages to ease the task of constructing a *Data Dictionary* to contain the necessary meta-data. Database languages are well suited for the task at hand because they are designed for the description and categorization of data. This data dictionary is the initial part of an SDD transfer, and the actual sensor data are serialized after the data dictionary as shown in Figure 8-1. An SDD transfer is composed of one data dictionary and a continuous stream of sensor data. An SDD transfer ends when a new data dictionary is transferred. We are proposing the SDD transfer method presented here as a robust mechanism for distributing ITS data to Information Service Providers (ISP's) (See the ITS National Architecture study for more information on ISP's [11, 12]. We are aware of two other efforts to produce standards for communication of transportation related data. They are the National Transportation Communications for ITS Protocol (NTCIP) [2] and the Transportation Network Profile of the Spatial Data Transfer Standard (SDTS) [3]. These standards were designed to meet significantly different needs, though both are designed to solve the problem of standardized data communication. The SDD paradigm presented here is related to aspects of both NTCIP and SDTS.

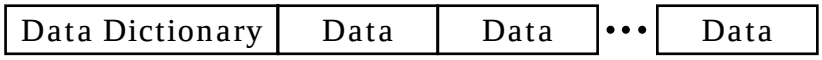


Figure 8-1. Data model for self-describing data transfers

The first related standard is the National Transportation Communications for ITS Protocol (NTCIP). NTCIP is a family of communications protocols (A, B, C, E) developed for real time communication between a master controller and field devices such as traffic signal controllers, environmental sensor stations, dynamic message signs, highway advisory radio, closed circuit TV, and freeway ramp meters [2]. For example, the class B protocol is designed for direct communication between a master controller and one or more field devices connected by a communications cable using low-speed data transmission (on the order of 1200 baud). The target application is the control of traffic signal management systems. The standard covers both the rules of transmission and the format and meaning of a set of standardized messages to be transmitted.

NTCIP is an extension of the Simple Network Management Protocol (SNMP) [13]. It extends the management information base hierarchical name-space defined (using ASN.1~[14]) for use with SNMP to include nodes devoted to the NTCIP. Groups of objects defined in this tree structured name-space are referred to as MIBs and represent the set of objects (in the object oriented design sense) that are needed to effect the desired control or information transfer operations. It is intended for many types of transportation devices, each with different database requirements. Control or data exchange is effected by modification of the objects in the MIB associated with the device(s) being controlled. This modification uses the get/set paradigm of the SNMP to change the values of objects in the MIB. The resulting action depends on the programming of the controlled devices. NTCIP is a control protocol which uses a globally agreed upon set of objects. To be effective, the control/

management software that uses the get/set operations needs to understand how the software within the devices reacts to changes in the MIB. So there must be an a priori agreement on software functions as well as object definitions.

From an NTCIP perspective, SDD transfers can be cast as a compound object for information transfer without a priori information about the data to be shared. Thus, SDD is targeted at information transfer and not control. The control function of NTCIP can be used to initiate or terminate SDD transfers. The data dictionary components and the actual data can be declared as objects and an ASN.1 compound object can be created. Changes in the MIB structure would then initiate or terminate an SDD transfer using the SNMP paradigms. The SDD transfer could be implemented either by viewing the MIB as a control that initiates an out-of-band data transfer as in [1] or the MIB could actually contain the components of SDD as objects. SDD transfers leverage NTCIP in that SDD has an agreed upon data description language that includes methods to describe and extract the elements of data from a data stream without the need for a great deal of a priori knowledge. For example, it is possible to create an application that would operate in a Java environment and that could obtain the methods from the data dictionary in the form of Java language elements and have those methods operate directly on the data stream, creating an automated transfer of data with only SQL and Java as the required a priori knowledge.

The second related standard is the Transportation Network Profile (TNP) of the Spatial Data Transfer Standard (SDTS) [3] that is designed for use with geographic vector data that has network topology. The SDTS allows transfers of spatial data that can be represented by vector objects that comprise a network or planar graph. The SDTS data types are *nodes* and *links* between nodes, each of which may have associated attributes. These associated attributes may be multi-valued, and therefore could be used to convey time-varying information associated with a node or a link.

The SDTS provides a mechanism for defining an external data dictionary module that need not be included in each transfer, thereby reducing the amount of overhead that must be devoted to sending a dictionary module for multiple transfers that use the same dictionary.

Though it was not designed specifically for the purpose of transferring large amounts of time-varying data, the SDTS could be used to transfer time-varying data from a set of remote sensors by representing the time-invariant information about the sensors (location, sensor type, etc.) as single-valued attributes of the nodes that make up the network, and the time-varying data would be represented as multi-valued attributes functionally dependent on time. If the time-dependent attributes are isolated into a separate table from the time-invariant information, it would be possible to send the module containing time-invariant data first and continue with the "open-ended'' time-varying module until no more time-varying data was desired at the receiving end. Though possible, transfer of large amounts of time-varying data using the SDTS has two disadvantages: 1) the data stream must be interpreted before transfer and placed into the appropriate attribute tables, possibly at a significant cost in required bandwidth and 2) the overall design of SDTS does not really include the notion of a transfer of indeterminate length. Our method makes a clear distinction between time-invariant data (the data dictionary contents) and time-varying data (the actual data stream), and allows for a more efficient treatment of the actual data stream.

8.3 Data Model

In the work presented here, the data to be transferred is modeled as having two components: 1) the data dictionary and 2) the actual sensor data. These two components, transferred serially, effect an SDD transfer. The Data Dictionary component is central to our self-describing data transfer method. In our model, the Data Dictionary is comprised of two parts: 1) Dictionary Schema and 2) Dictionary Contents. These two parts provide the necessary description of the data to make it useful to a client and are described in the next sections.

8.3.1 Dictionary Schema

The first part of our Data Dictionary is the *Dictionary Schema*. This is meta-data that specifies the schema of the data description (e.g. a sensor has a name, position, and units of measure and the position is specified in latitude

and longitude). The dictionary schema is a provider-defined database schema written in a subset of Entry Level SQL-92 [15]. We chose Entry Level SQL-92 because it is fully relational and the relational model can represent an arbitrary set of data [16]. The SDD model presented here allows for the construction of any schema that SQL allows, and this guarantees a powerful data-description language. In the Dictionary Schema, the data provider should include sufficient information about the actual data to allow a recipient to interpret that data. Because the sufficiency of the data dictionary is dependent on its author, it is clear that the data dictionary concept *allows* for self-describing data transfer but does not *ensure* that any given data transfer is in fact self-describing. It would seem difficult, if not impossible, to make such an assurance in an automated system. It is, however, possible to automate the verification that the schema provided conforms to the data description language.

As part of the SDD transfer method, we have created a schema parser which is used to verify the data dictionary schema definition. The language accepted by the parser is a subset of entry level SQL-92 that allows definition of schemas, tables, etc. but does not include any of the query processing facilities of a complete database language. Our intent in defining the schema language is to provide sufficient power for the definition of a data dictionary while simultaneously making the language simple enough that it is easy to learn.

The schema parser is used in two ways by our self-describing data transfer protocol. First, the sending application uses the parser to verify that a user-provided schema definition is valid. The second use of the parser is the construction of a parse tree that is subsequently used to verify that the dictionary contents are compatible with the defined dictionary schema. The receiving application uses the parser again to verify that the received dictionary schema is a valid one and constructs a parse tree that is used to create and verify dictionary contents file.

The structure of the schema language is such that the maximum depth of the parse tree is four. Further, at each level, the nodes are of a specific type. The four types, in order of increasing depth in the parse tree are catalog, schema, table, and column. For entry level SQL-92, the catalog and schema nodes are not really needed since only one of each can exist. A single schema node could be used as the root of the parse tree. We chose the 4-level tree implementation to make it easier to extend to higher levels of SQL-92 conformance (which allows multiple schemas and catalogs).

8.3.2 Dictionary Contents

The second part of the data dictionary is the *Dictionary Content*s. The dictionary contents is the information about the actual data stream that can be used to construct a database describing the static information about the sensors for this data transfer. This is the component that contains particular values for the description of each sensor (e.g. sensor one is at 122.23 degrees longitude, 47.21 degrees latitude and measures rainfall in inches). We define a *Contents Language* and associated parser to facilitate verification that the schema contents are compatible with the schema into which they are to be placed. A contents parser recognizes the language used to describe the contents of the data dictionary. The language is designed to allow specification of the table and columns into which a set of data tuples are to be inserted. The contents language is fairly simple. A *data dictionary file* consists of a series of *table entries*. Each table entry is of the following form:

```
TABLE <table name> COLUMN (<column name 1>, ... , <column name n>) <data for
column name 1>, ... , <data for column name n> ;
 ... one tuple for each row to be inserted into the table <data for column
name 1>, ... , <data for column name n> ; .
```

The parser ensures this format and ensures that the type of data supplied in each tuple matches the data type declared for its corresponding column. On the sending end of a data transfer, the contents parser is used to verify that the contents file supplied by a data provider is compatible with the dictionary schema in use. On the receiving end, the contents parser is used to verify that the data in the contents file are compatible with the schema during SQL command generation.

8.3.3 Data Transfer

The overall architecture of our system for self-describing data transfer is shown in Figure 8-2. This structure is independent of the actual data stream involved. The data transfer is a serialized stream divided into frames. The first frame of a transfer contains the Dictionary Schema. The second frame contains the Dictionary Contents; subsequent frames contain the most recently available set of data from the data source. The implementation of a self-describing data transfer takes the form of a transmitter and a receiver.

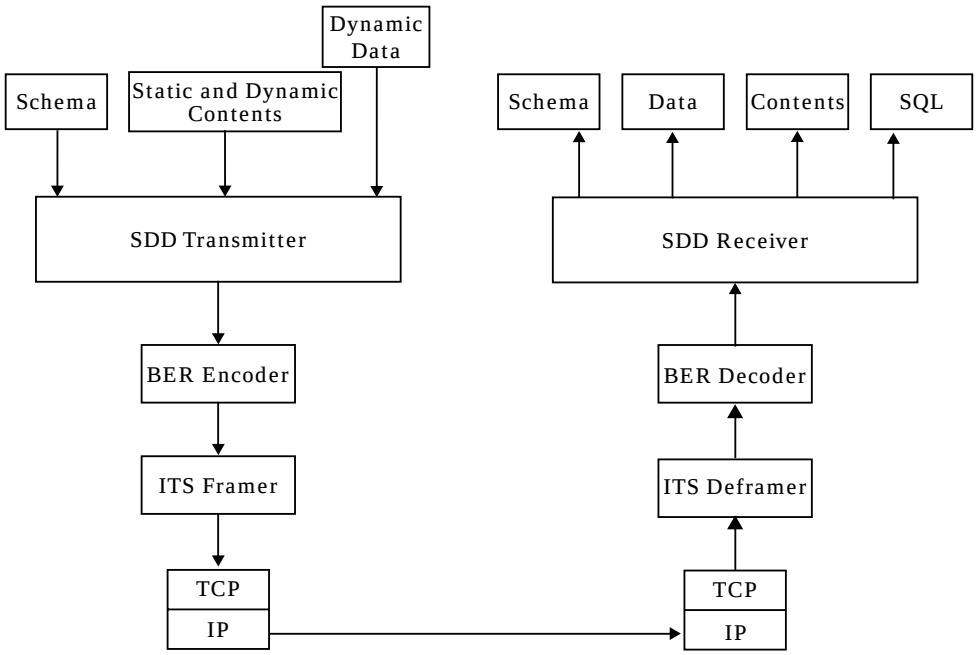


Figure 8-2. An overview of the structure of our system for self-describing data transfers

8.3.4 Transmitter

Figure 8-3 shows the components necessary to construct and transmit a stream of self-describing data. The SDD transmitter verifies the compliance of the schema using the schema parser, which creates a parse tree as a byproduct of the compliance check. This parse tree is then used by the contents parser to verify that the contents complies with the schema. If both the schema and the contents are in compliance, they are serialized in the order schema, contents, and the actual sensor data. Transfer of the dictionary and data between computer systems is accomplished by encoding the data according to the Basic Encoding Rules (BER) defined for the ASN.1 standard [17]. As a block of data is received, it is encoded and sent to all clients using the redistributor methodology detailed in [1]. In our application, the data is a stream of bytes, and the information is extracted using algorithms specified in the Data Dictionary. This mechanism allows for a very general transfer that is easy to encode, but which requires programming effort to be devoted to data extraction by the transfer recipient.

BER encoding of the self-describing data stream involves three types: data, schema, and contents. Each of these types is encoded according to the BER standard (ISO 8825-1) [17], with a type, length, and value.

We use the ASN.1 "Application" class with the "primitive" encoding, using tag numbers 1, 2, and 3 to denote schema, contents and data, respectively. Our identifiers are therefore encodable in one octet. We encode the "schema" and "contents" types as IA5 strings, and the "data" type is unchanged during encoding. An ASN.1 type declaration of our types is:

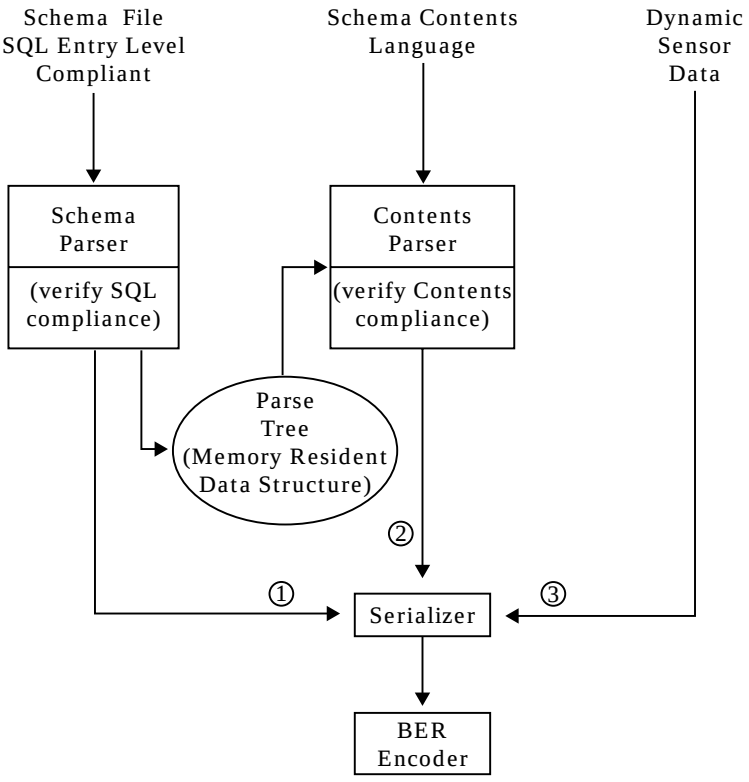


Figure 8-3. Self-describing data transmitter

```
DictionarySchema ::= [APPLICATION 1] IA5String
DictionaryContents ::= [APPLICATION 2] IA5String
Data ::= [APPLICATION 3] OCTET STRING
```

The tag number needs to be unique across the family of protocols using these three data types. For example, if used with the NTCIP family of protocols, the NTCIP standards document would need to dedicate three types to an SDD transfer facility. The serializer in Figure 8-4 sends a type, length, and value to the BER Encoder. The BER Encoder encodes these values as the appropriate header, length bytes, and contents bytes according to the ASN.1 definition above.

The SDD Receiver is a client application that converts from the data transfer stream format back to three data sets: schema, contents, and data. The structure of the receiver, as shown in Figure 8-4, is parallel to that of the transmitter in Figure 8-3. The BER decoder takes an SDD data stream as an input, BER decodes the data stream, and provides a decoded serial data stream that has the structure described in Figure 8-5. The BER Type Demultiplexor receives a serial data stream from the BER Decoder, and it uses the BER type field to distribute the schema and contents to the appropriate parser. The parser components of the receiver are identical to those of the transmitter. The schema component is sent to a schema parser that verifies SQL-92 compliance and creates a parse tree for use by the contents parser. The contents are sent to the contents parser, which verifies the contents against the schema. The outputs of these two parsers are the verified schema and contents used to describe the dynamic sensor data. With the arrival of the data dictionary schema, data dictionary contents, and the sensor data, a self-describing data transfer is complete.

In our implementation, we have added an SQL generator, as shown at the top of Figure 8-6. The SQL Generator creates a series of SQL INSERT commands from the dictionary schema and dictionary contents. Each data tuple in the contents file will be represented by an INSERT statement in the SQL output file. The commands, when

used as input to an SQL database engine, will create and fill a database that instantiates the data dictionary. We include this step as a practical matter for the engineering users of the SDD paradigm, so that the result of an SDD transfer is a data base with which an engineer can interact and a binary stream of dynamic sensor data. The practicality of this paradigm is demonstrated using an ITS example in the next section.

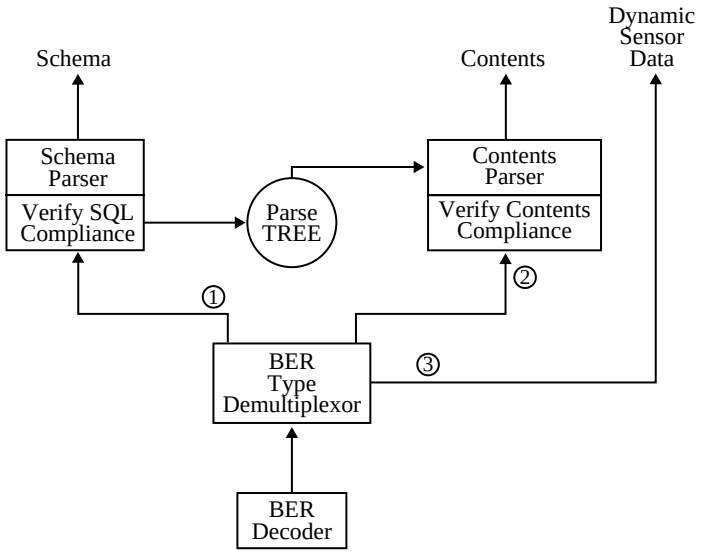


Figure 8-4. SDD receiver

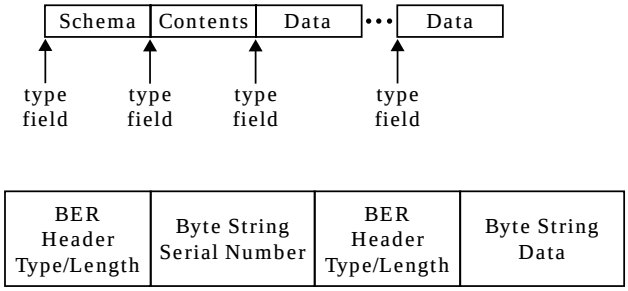


Figure 8-5. Structure of our serialized data stream for self-describing data transfers

8.4 ITS Example

The Washington State Department of Transportation (WSDOT) operates a Traffic Management System (TMS) to collect data from traffic sensors in the central Puget Sound region, and this TMS traffic data is the focus for this ITS application of SDD. The data from the TMS consists of three parts: 1) sensor data which is binary data representing measures of traffic conditions (e.g. speed, flow, occupancy), 2) information about the location and type of each sensor (e.g. mile marker, latitude, longitude, calibration), and 3) lists of presently available sensors. The sensor data comes from inductance loop detectors placed at approximately 3000 locations in the Seattle metropolitan area, and the list of sensors/detectors presently available changes slowly with time (e.g. every few hours). This dynamic list of available sensors makes up the dynamic component of the data dictionary contents. The name, location, measurement type, etc. make up the static component of the data dictionary contents.

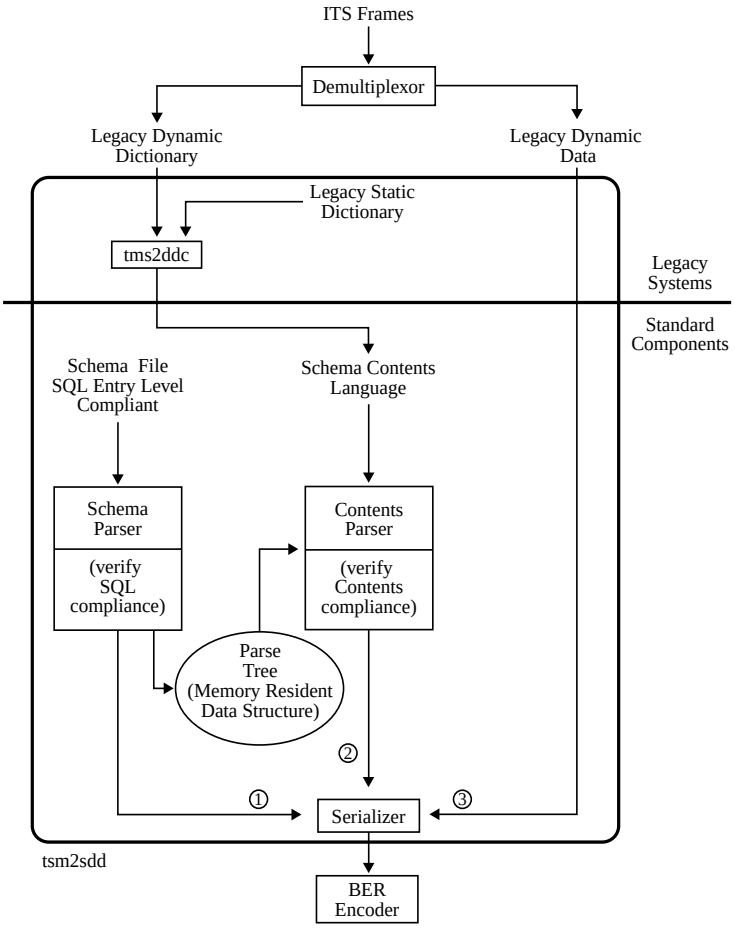


Figure 8-6. An overview of the structure of our example application for self-describing data

Figure 8-6 provides details about our "*tms2sdd"* application, which converts legacy TMS data to our self-describing transfer format. The application can be divided into a "legacy'' component and a "standard'' component. The legacy component is dependent on the specific data source; the standard component does not change from one source to another. As in the generic transmitter case, the self-describing dictionary contents file is verified against the dictionary schema by the standard component of *tms2sdd.* The schema is verified by the Schema Parser, which constructs a memory-resident parse tree that is used by the Contents Parser during the process of parsing and verifying the dictionary contents file. If verification of both files succeeds, they are transmitted to the BER Encoder.

8.5 SDD Applications

The Self-Describing Data (SDD) applications programming interface (API) of the ITS Backbone is depicted in Figure 8-7. The overall backbone design includes transmitters, operators, and receivers, stacked into three functional layers: Domain, SDD, and Frame, as shown in Figure 8-7. This paper describes the software presently available that implements the receiver portion of the API (software and documents available at http:// www.its.washington.edu/bbone). The receiver software included is represented schematically by the SDD 2.0.0 column to the right of the Receivers in Figure 8-7. The software is based in an object-oriented paradigm that maps into the three functional layers.

The ItsFrameReceiver (see Figure 8-7) is a class that receives data from a network socket connection and produces ItsFrame events. It is initialized and connected to an SDD data source using a host name and data port

number. Once the connection has been established, a number of ItsFrame objects are transmitted to the receiver. These are validated, serialized, and made available to registered classes via the itsFrameReceived callback method (see Figure 8-7), which takes an ItsFrame as input.

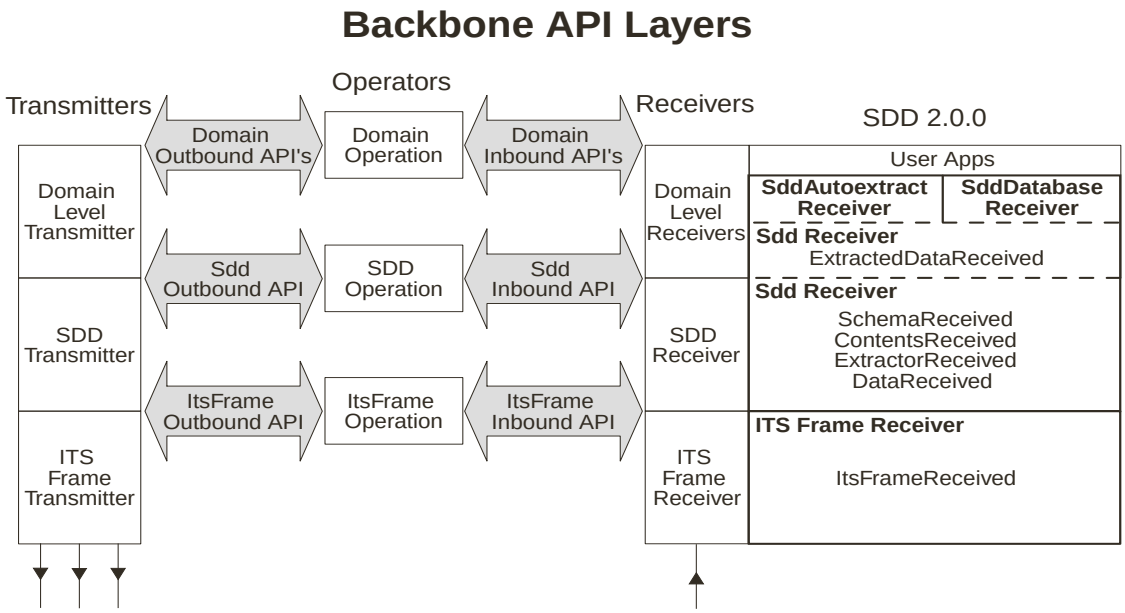


Figure 8-7. SDD Aplication programming interface

The SDD layer produces four individual types of data:

1. Schema: an object containing an SQL2 compliant data-definition language describing the data stream as a collection of tables.
2. Contents: an object containing meta-data values to be inserted into one or more of the tables defined by the Schema.
3. Extractor: a compressed JAVA jar file containing a data factory class that can convert the raw binary into tabular objects defined by the data tables (those tables ending with "_DATA") in the Schema.
4. Data: the raw binary data.

The domain level of the API allows users to access Extracted Data. Extracted Data is the result of expanding the raw binaries into tabular objects corresponding to the data tables defined in the Schema. These events are produced in the SddReceiver by taking the latest Extractor and applying it to each of the Data events in the incoming stream.

Two example applications are available. The first, SddAutoExtractReceiver, (the control panel is shown in Figure 8-8) allows the user to connect and get data from *any* Sdd data source. TMS Loop data is available at port 8411, and Automatic Vehicle Location (AVL) data is available at 8412. The user is required to set the host and port for the SDD data source, and this application produces output files containing the Schema, Contents, Extractor, Data, and ExtractedData. The user can specify which of these files to produce using check boxes.

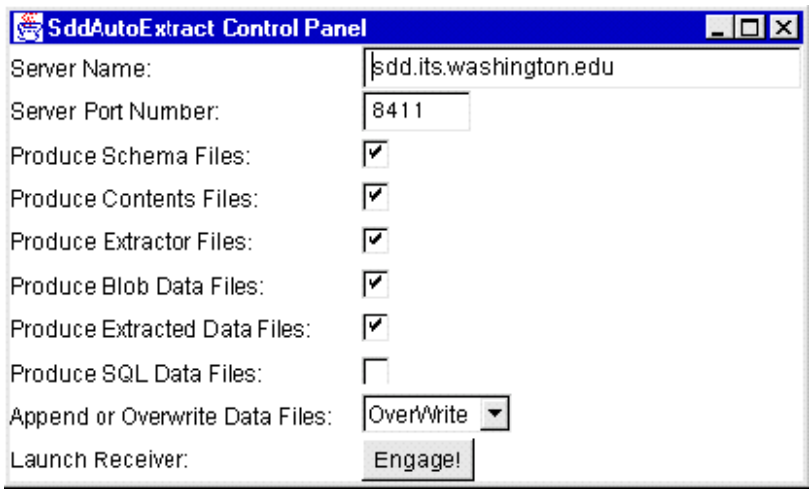


Figure 8-8. Control panel for SddAutoExtractReceiver

The second application, SddDatabaseReceiver, loads Schema, Contents, and ExtractedData directly into a database, leveraging Java DataBase Connectivity (JDBC) capabilities. To run this application, a database engine with remote access capability and its corresponding JDBC driver is required. This application produces the control panel shown in Figure 8-9. The user specifies the SDD data source host and port, the JDBC class and URL descriptions, as well as the user name and password. The application contains several examples of class name and URL templates from Oracle and Sybase. A meta-data toggle informs the application to load the Schema and Contents into the database. When this toggle is set to off, only extracted data is loaded and the meta-data is ignored. The time required to load extracted data will vary with the number of inserts, the speed of the database host, and the speed of the connection from the client running the application to that host. If the ExtractedData events arrive faster than the host can insert them, "overflowing" events will be dropped.

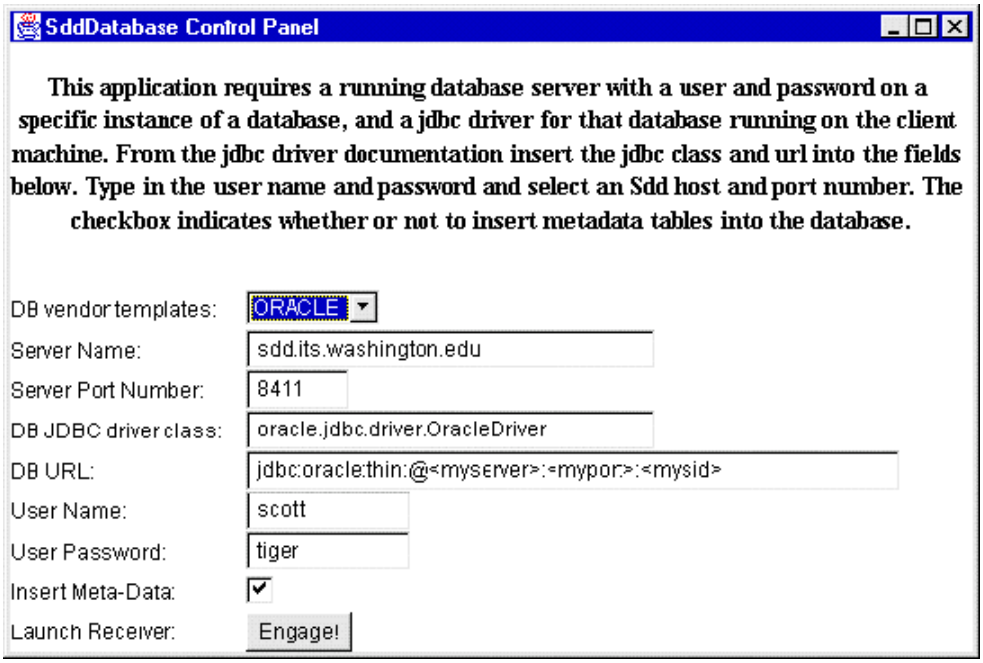


Figure 8-9. Control panel for SddDatabaseReceiver

The SDD methodology described here is being deployed for ITS traveler-information applications in the Seattle metropolitan area in the form of the ITS Backbone for the SmartTrek Model Deployment. Information Service Providers use the SDD receiver to connect to the regional ITS backbone and obtain both the Traffic Management data just described or Transit Coach information.

8.6 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, 1996, pp. 163-180.

2. NTCIP Steering Group. *National Transportation Communications for ITS Protocol (NTCIP): A Family of Protocols*. Published on the World Wide Web at http://www.ntcip.org/abstracthtml/family01.html, May 1996.

3. NTIS. *Spatial Data Transfer Standard: FIPS 173-1*. National Technical Information Service (NTIS) Computer Products Office, 5285 Port Royal Road, Springfield, VA 22161, (703)487-4650 (Specify FIPSPUB 173-1, parts 1 through 4, when ordering). June, 1994.

4. Hall, S.R. The STAR File: A New Format for Electronic Data Transfer and Archiving. *Journal of Chemical Information and Computer Sciences*, Vol. 31, No. 2, May 1991, pp. 326-333.

5. Wideman, G. "Streamlining Experiment Data Manipulation With Psychology Experiment Data Interchange Format (PXDIF). *Behavior Research Methods, Instruments, and Computers*, Vol. 23, No. 2, May 1991, pp. 288-291.

6. Allen, F.H., J.M. Barnard,, A.P.F. Cook, and S.R. Hall. The Molecular Information File (MIF): Core Specifications of a New Standard Format for Chemical Data. *Journal Chemical Information and Computer Sciences*, Vol. 35, No. 3, May-June 1995, pp. 412-427.

7. Mark, L. and N. Roussopoulos. Information Interchange Between Self-Describing Databases. *Information Systems*, Vol. 15, No. 4, 1990, pp. 393-400.

8. Hall, S.R., F.H. Allen, and I.D. Brown. The Crystallographic Information File (CIF): a New Standard Archive File for Crystallography. *Acta Crystallographica, Section A (Foundations of Crystallography)*, Vol. A47, Pt.6, 1 November 1991, pp. 655-85.

9. Hall, S.R. and A.P.F. Cook. STAR Dictionary Definition Language: Initial Specification. *Journal of Chemical Information and Computer Sciences*, Vol. 35, No. 5, September-Octobert 1995, pp. 819-825

10. Johnson, J.A. and F.C. Billingsley. A Standard Method for Creating Self-Defining Data Structures for Information Archive and Transfer. *Digest of Papers. Tenth IEEE Symposium on Mass Storage Systems. Crisis in Mass Storage (Cat. No. 90CH2844-9)*, ed. by Friedman, K.D. and B.T. Olear. IEEE Computer Society Press, Washington, DC, USA. Monterey, CA, USA, 7-10 May 1990, pp. 26-32.

11. Joint Architecture Team, Loral Federal Systems, Rockwell International. *National ITS Architecture: Theory of Operations*. Published on the World Wide Web at http://www.itsa.org/Archdocs.nsf/Files/Theory/$file/Theory.pdf as Adobe Acrobat PDF file, October 1995.

12. Joint Architecture Team, Loral Federal Systems, Rockwell International. *National ITS Architecture: Physical Architecture*. Published on the World Wide Web at http://www.itsa.org/Archdocs.nsf/Files/Physical/$file/Physical.pdf as an Adobe Acrobat PDF file, October 1995.

13. Stallings, W. *SNMP SNMP-2 and RMON: Practical Network Management, 2nd Ed*. Addison-Wesley Publishing Company, 1996.

14. ISO. *ISO/IEC 8824-1 Information Technology - Abstract Syntax Notation One (ASN.1): Specification of basic notation*. ISO/IEC Copyright Office, Case postale 56, CH-1211 Geneva 20, Switzerland, 1995.

15. ANSI. *American National Standard Database Language SQL, ANSI X3.135-1992*. American National Standards Institute, 1992.

16. Codd, E.F. A Relational Model of Data for Large Shared Data Banks. *Communications of the ACMe*, Vol. 13, No. 6, June 1970.

17. ISO. *ISO/IEC 8825-1 Information Technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)*. ISO/IEC Copyright Office, Case postale 56, CH-1211 Geneva 20, Switzerland, 1995.

9. Seattle MMDI Integration Case Study: ITS Information Backbone [15]

9.1 Introduction

This report describes the results of a case study performed on the Intelligent Transportation System (ITS) Information Backbone (I^2B) component of the Seattle Metropolitan Model Deployment Initiative (MMDI) project. After describing the goal and scope of activity for this research, background information on the I^2B will be provided. Following this information will be the results describing the benefits and lessons learned from the project.

9.2 Goal

The case study investigated the effects of the I^2B on integration among ITS Advanced Traveler Information System (ATIS) activities, participants, and players. A key component of the study was the determination of the nature of any increased benefits or reduced costs associated with the integration. This was accomplished primarily through interviews with key participants in the project, in addition to a review of available literature and reports.

The study provides information on integration resulting from I^2B which would not be gained from a quantitative evaluation. Since integration from a project of this type involves many non-quantifiable factors/benefits, and because many values are difficult to quantify in absolute terms, the study focused on qualitative information. Anecdotal or informal information may be more useful to other agencies interested in learning key lessons from the deployment of the I^2B. In addition, anecdotal information may be among the few types of information which have been available for some types of ATIS benefit measures (Mason 1998).

9.3 Scope of Work

The scope of the study is limited to current, existing ATIS activity, infrastructure, processes, participants. Any efforts which are still in the planning stage or which have not been deployed were not considered. Although this eliminated many elements of the project from consideration, it was important to make when possible a distinction between actual benefits, as in Apogee/Hagler Bailly (1998), and predicted benefits which may or may not occur in the future. End user level effects were considered to be outside of scope of evaluation, as the I^2B primarily operates at the organization level. The benefits to the actual travelers will be evaluated in other efforts. As a background, some benefits resulting from ATIS projects are discussed in Apogee/Hagler Bailly (1998, pp. 6-7), and Salwin (1996, pp. 7-8).

9.4 Research Approach

The approach used for this study (Novak 1998) was proposed by Dave Novak of the Virginia Tech Center for Transportation Research. The general approach, designed for MMDI case studies and tested on the Video Sharing in San Antonio project, involves the collection of benefits information from various sources. This information is then consolidated, allowing ideas and lessons to be drawn, but avoids the prediction of outcomes in other situations. The first component of data collection for this study involved the review of available literature, reports, and other materials on the I^2B. This provided both foundation background information and benefits information. The second and major part of the data resulted from telephone interviews conducted with key participants in the Seattle MMDI related to the I^2B. These interviews attempted to gather anecdotal

[15] Draft report preared by John Collura, James Chang, Virginia Tech,Center for Transportation Research - ITS Research Center of Excellenc, Falls Church, Virginia and Mark Carter, Science Applications International Corporation,McLean, Virginia, September 9, 1998

information, including perceptions of benefits, problems, and other issues relating to the I²B. The reminder of the report describes the results of the information collection.

9.5 Background

In order to understand the benefits associated with the I²B, it is first necessary to have a basic grasp of the I²B structure and operation. Conceptually, the I²B is an information system design. It specifies how participants organize information flows, consisting of gathering, processing, and dissemination steps, from the source to the ultimate users. From a functional perspective, the I²B acts as a collector and distributor of information. (Bradshaw 1998; see Figure 9-1) It receives information from various contributors, and redistributes this information to processors, who process the information, add value, and give it to the traveler in the desired form (System Engineering Requirements Specification, p. 10). The actual technical components of the I²B consist of a series of geographically distributed computers, located at the University of Washington (UW) and other sites including the Washington State Department of Transportation (WSDOT) TSMC and the King County Metro Transit Operations Center (System Engineering Requirements Specification 1997, p.75). These computers host processes which interact and collaborate with each other, using the Internet as the means of communication (Dailey 1998).

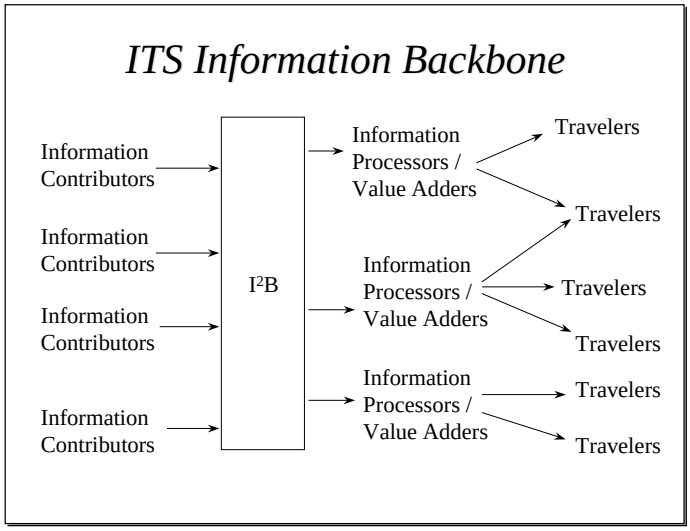


Figure 9-1. ITS Information Backbone design concept

An analogy of the I²B concept could be the operation of the AP Newswire. The contributors of information would consist of AP reporters, who are geographically distributed and collect local information. The I²B would be represented by the AP editors and the distribution "wire" to newspapers. The editors maintain the standard of the AP, while the "wire" performs the actual distribution of stories to the newspapers and other media. The processors would be the local newspapers or media outlets which take the stories or other information (e.g. photos) from the "wire" and add value through republishing, summarizing, or producing derivative works. They then provide the value-added information to the ultimate users in the form they desire, such as a newspaper on the newsstand.

The role of the I²B in integration may be shown through the Oak Ridge Integration Schematic (ORNL undated). At present, the I²B incorporates the functions designated as #10, #14a, and #14b on the Oak Ridge Schematic (see Figure 9-2). The link between freeway management and regional multimodal traveler information may be considered to represent in this case the provision of freeway loop information from WSDOT to the I²B. The links between transit management and regional multimodal traveler information may represent the automatic vehicle location (AVL) data on buses generated by King County Metro and sent to the I²B.

9.6 Existing Participation

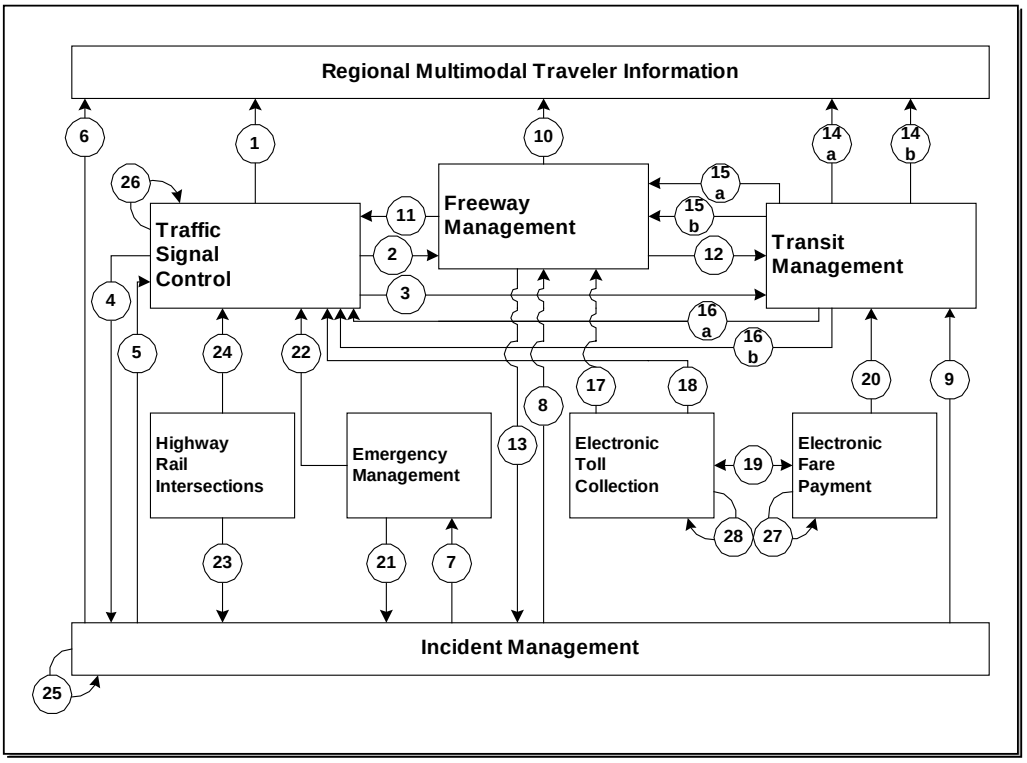


Figure 9-2. Oak Ridge schematic

As mentioned earlier, the study focused upon existing components and operation of the I²B. The existing state of the I²B as considered by this study is described as follows (see Figure 9-3). WSDOT currently contributes freeway loop data to the I²B, while King County Metro contributes bus location information. The administration of the I²B is performed by the University of Washington (UW), which also provides software tools for

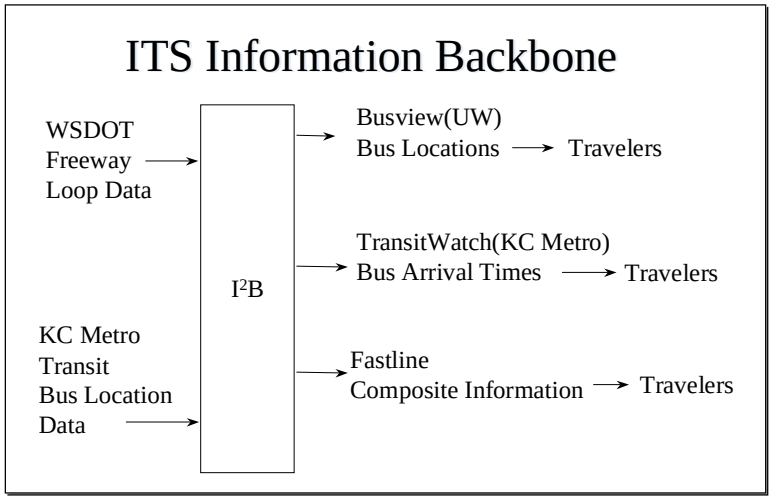


Figure 9-3. Existing state of backbone

connecting to the I²B. UW also operates Busview (see Figure 9-4), a service which uses the World Wide Web (WWW) to display KC Metro bus locations graphically. King County Metro operates the other transit-specific application, TransitWatch (see Figure 9-5), which displays estimated bus arrival times at key transit centers. Fastline uses both highway and transit information to provide value-added composite information to its customers on handheld computing devices. The previously described services represent the currently deployed applications; however, many other applications, such as the Seattle Center Parking Information System, are slated to use the I²B. Using the same information system and communications infrastructure, a tremendous amount of information for the future can be set up (Patton 1998).

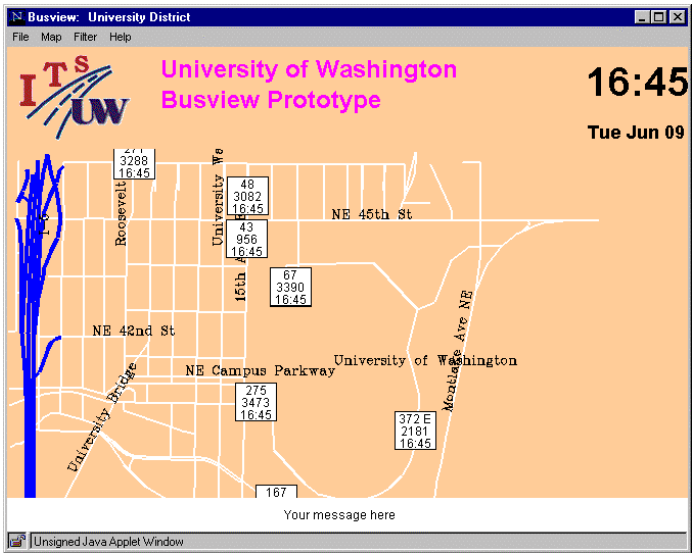


Figure 9-4. Busview protoytpe screen shot

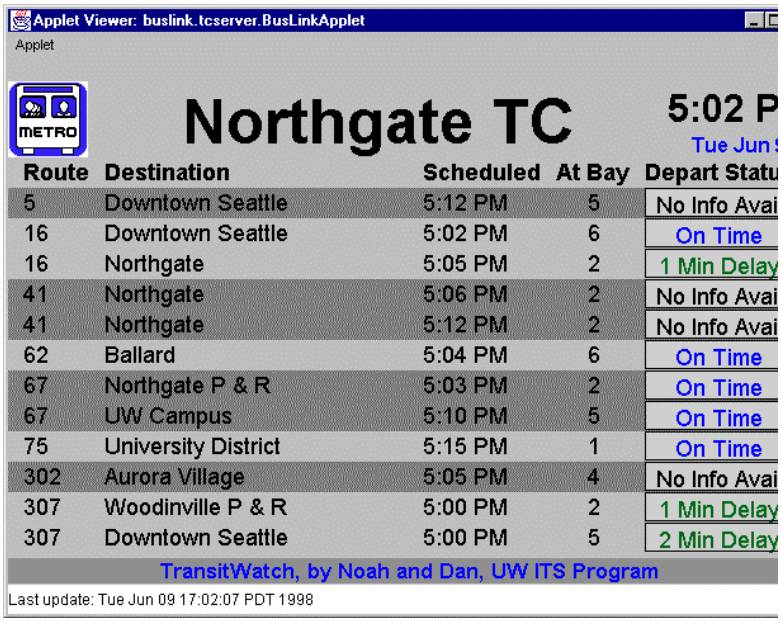


Figure 9-5. TransitWatch prototype screen shot

9.7 Results from Literature Review and Interviews

9.7.1 Information System Design

The I²B utilizes distributed communications infrastructure consisting of the Internet for multiple access methods. I²B users may connect to it via an appropriate medium supported by Internet Service Providers, including ISDN, T1, etc. The use of the Internet allows a standardized access point rather than custom communication links between parties in an information flow. From a communications perspective, it would only be necessary to establish an Internet connection in order to participate (Dailey 1998); negotiations with other parties would not be necessary. In this sense, the I²B design allows the Internet to serve as a key piece of shared infrastructure.

The distributed nature also applies to the issue of coordination of responsibility. Multiple contributors and processors have only to take responsibility for using data for their own needs. If processing is beyond the scope of a contributory public agency, the cost would be the responsibility of the processor (System Engineering Requirements Specification, p.74), though secondary agreements are always possible. Agencies without the staff resources or orientation toward ATIS outreach may still be able to provide useful data (Dailey 1998). In keeping with this notion of facilitation of data contribution, the focus with the I²B has generally been a voluntary one rather than one of command and control (Cima 1998).

Another feature of the I²B is the openness of participation. A potential participant would have a relatively easy time connecting to the I²B, as UW provides a software toolkit for connection, available on the World Wide Web (Dailey 1998). As mentioned before, the standardized access via the Internet allows a party to connect without any requests for a custom communication link or process. For example, WSDOT used to deliver flow maps via telephone to processors after setting up an arrangement; now, the data is shared with all interested parties by virtue of its direct availability via the I²B (Briglia 1998). Also, although UW administers the I²B, no one party controls the backbone. A public agency such as the City of Lynwood may benefit from traffic information from its surrounding area in order to plan and act on regional interactions, yet would not have to give up control over its own traffic signal systems (Briglia 1998).

Again, since the I²B uses the Internet as the communications infrastructure, many geographic constraints which existed in the past have been removed. Potential participants may develop, setup, and even operate applications from any geographic location via the Internet. Though in many cases the party desires to have an operating presence in the area of interest, the development effort may sometimes be accomplished more efficiently or effectively at an alternate site. For example, during the SWIFT project, which contained precursor elements of the I²B, Seiko did much of the development work in their existing office in Oregon (Dailey 1998).

The I²B has a uniform data distribution method, known as “self-describing data” (SDD). With this concept, descriptive information about the data is sent alongside the data itself so that the format and other necessary information are always available (SDD for Dummies, 1997, p. 1). As a result the contributor agency is responsible for providing the data formats, but may reduce the amount of resources expended on multiple requests for information from different parties.

9.7.2 Costs Estimates

Original funding for design and demonstration of the I²B was listed at $165,000 (UW ITS Projects WWW). The SDD element involved funding of $245,500 (UW ITS Projects WWW). An estimated half of the $125,000 Data Fusion project, 10% of the $1,131,140 UW SWIFT element, and 10% of the $1,577,017UW Smart Trek element (UW ITS Projects WWW) were added to result in a rough estimate of total deployment cost of $750,000. Annual costs for I²B deployment and operations and maintenance currently are allocated an amount of $450,000 from state funds (Dailey 1998). Johnson and Chen (1998, p.27) estimate that 10-15% of the project cost would be a rough estimate for annual operating and maintenance cost, resulting in about a $100,000 annual operations and maintenance cost. However, because of the interdependence and interaction between projects under SWIFT and Smart Trek, it is difficult to clearly separate which elements are part of the I²B and how costs should be allocated. Also, note that these costs were estimated for the I²B itself, and not for components used by

those connecting to the I²B. These costs are substantial but are not really "necessary" for the I²B to operate. Since the I²B is essentially infrastructure, the user costs should likely be accounted for under the individual applications. Due to differences in functionality provided through the associated applications, it is difficult to make a direct comparison of costs between the I²B and alternative systems (Dailey 1998) such as the one in place in San Antonio. For example, an alternative system design (see Figure 9-6) would have integral applications for which the costs of the application and the costs of the information system may not be easily separated.

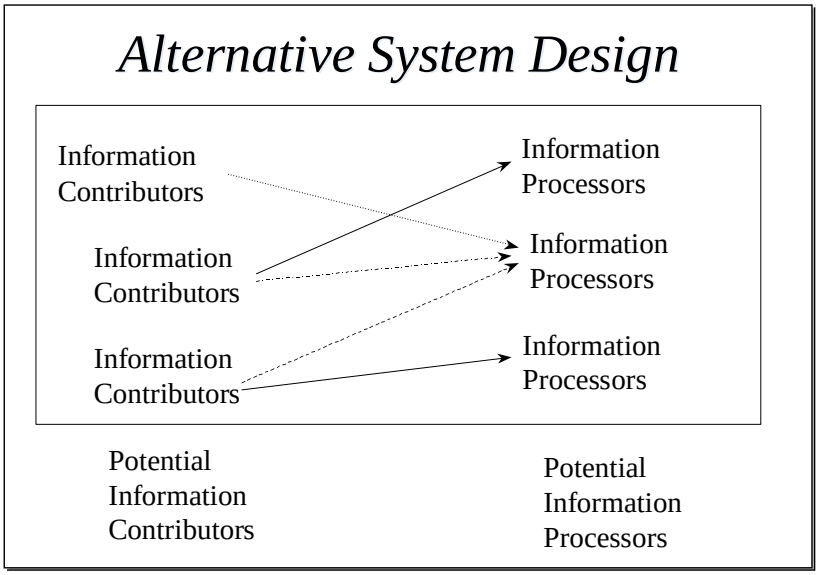


Figure 9-6. Alternative backbone system design

9.8 Benefits to Contributors

9.8.1 Level of Effort

With the I²B, contributors can provide and share data gathered for their own existing or internal needs with little additional effort and constraint. In most cases, the public agencies act as contributors, and have generally decided that it is in their own best interest to get information to travelers (Briglia 1998), though agencies may prefer other parties to accomplish the task (Bradshaw 1998). In the case of KC Metro, the automatic vehicle location data on its bus fleet was already being collected for fleet management purposes. This information is provided in real-time to the I²B without any further processing and with minimal effort on the part of KC Metro (Bradshaw 1998).

9.8.2 Low Overhead

The I²B has also benefited contributors by reducing the burden of supporting processors using the data. Contributors and processors need only to deal with the backbone and not necessarily each other. Previously, a common way for a provider to get information from WSDOT was to establish a connection, which required staff time and resources from WSDOT. By using a standardized access method, staff time to respond to data users and resources used to establish connections would be reduced for WSDOT (Briglia 1998). For example, summary flow maps used to be delivered by telephone to processors; now detailed raw data may be retrieved without the interaction of WSDOT staff (Briglia 1998). KC Metro has maintained low overhead in contributing information to the I²B by leaving a computer on its internal network which captures AVL information and redistributes it to the I²B without user intervention (Bradshaw 1998).

9.8.3 Geographic Distribution

Contributors are afforded the freedom to provide data independent of their geographic location. Since the Internet serves as the shared infrastructure for communication both to and from the I²B, any connection to the Internet with appropriate characteristics could be used. There would be no need to send data back to a central site unless it was desired for the contributor's own purposes. This permits geographically distributed data sources to be directly connected to the I²B. WSDOT has its connection located at the TSMC, while KC Metro Transit has a computer on its internal operations center network. A potential application which would benefit from this geographic freedom would be the gathering of visual information in the form of digital images on-site at a highway incident (Patton 1998).

9.9 Benefits to Processors

9.9.1 Geographic Distribution

Similar to contributors, processors also benefit from geographic freedom. Processors may develop applications off-site (Dailey 1998) and have reduced constraints in locating operations. Free software tools to interact with the I²B are available on the World Wide Web (WWW) at UW and may be retrieved from anywhere on the Internet. In addition, retrieving information from the I²B from multiple sources would still involve only the single connection to the Internet.

9.9.2 Easier Access to Information

As mentioned before, public information which has been gathered for existing uses may be more likely to be shared as a result of the ease of contributing to the I²B. This in turn benefits processors who rely on the availability of information for which they may add value. Had the I²B not been in place, it is likely that KC Metro would not have shared the AVL data with a company like Fastline, due to institutional constraints (Bradshaw 1998). The I²B allowed KC Metro to give everyone equal access to data thus avoiding the potential need to bid out on a contracted basis (Bradshaw 1998). The I²B also provides a single "one stop source" from which concentrated data can be accessed (Lundburg 1998). If processors had to go to multiple information sources, it might be cost prohibitive or otherwise infeasible (Patton 1998). In some cases, such as Fastline, the I²B allows a processor to work without needing to interact with contributor agencies (Lundburg 1998).

9.9.3 Composite Information

Another benefit to processors resulting from the I²B is the ability to develop composite information services based upon multiple information sources. Processors can easily draw upon multiple sources and types of information, including those from different jurisdictions or different modes. Fastline is able to provide both freeway congestion information and transit bus information to its customers on the same portable computing device (Lundburg 1998). This type of information may allow travelers to make decisions based upon a better understanding of the alternatives available. In addition, the I²B facilitates the ability to perform fusion of data streams, producing additional value-added data streams which did not exist before (Dailey, Haselkorn, and Meyers undated, p. 3).

9.9.4 Avoiding Startup Costs

Processors desiring to enter the market benefit from the I²B because it lowers startup costs. If the processor does not have an existing operation, the I²B can provide a large advantage in providing access to public data rather than having to attempt to gather it independently (Letshaw 1998). In some cases, such as the situation Fastline faces with regard to bus location information, the processor relies on the public agency to provide data; if KC Metro did not provide it, Fastline would not really have been able to gather it independently (Bradshaw 1998). Without the I²B, potential processors would not be as likely to start due to high infrastructure costs to gather

information (Patton 1998). The provision of the software tools to operate with the I²B by UW has also given processors an advantage in saving time (Lundburg 1998).

9.10 Other Issues

It has been suggested that the I²B may in fact transfer costs from contributors to processors rather than reducing the costs (Cima 1998). Support costs for contributors may decrease, but processors may have to get help if they cannot connect; but the added value of the information would generally be gained by the processors and not the contributors. Another issue discussed is potential barriers to private sector companies acting as contributors. It was conjectured that since information is of value, private for-profit entities may not find contributing data to be of competitive advantage as it may benefit competitors (Letshaw 1998). This may be a factor in the lack of private participation on the contribution side, though the organization of the I²B may be more geared to the redistribution and adding of value to public data rather than private data.

9.11 Summary / Lessons Learned

A key benefit of the I²B is to facilitate the information flow from contributor agencies to the ultimate user, the traveler. It reduces the barriers to participation, thus increasing both the availability of data and the likelihood that a party will add value to it and distribute it to travelers. The project has provided for a standardized communications infrastructure and software tools to manage interactions. The I²B also facilitates the integration of multiple data sources, resulting in composite applications which may not have been feasible before. In addition the I²B has provided a division of responsibilities, allowing contributors to focus on providing data without an excessive burden of support, and encouraging processors to add value with the potential for profit. The likely transfer of support costs from contributors to processors seems consistent with the potential profits on the processor side of the I²B and the free access to raw data. This access to existing data allows parties who see potential saleable added value to develop applications and gain any profits from realized value. This may encourage market-driven activities as the private sector for-profit entities have potential opportunities resulting from the free data but also have potential competition to add value.

9.12 References

1. Apogee/Hagler Bailly, *Intelligent Transportation Systems: Real World Benefits*, Federal Highway Administration ITS JPO Report FHWA-JPO-98-018, January, 1998.

2. Bradshaw, Catherine, King County Metro Transit, Telephone Interview on June 30, 1998.

3. Briglia, Pete, Smart Trek Project Manager, Telephone Interview on July 16, 1998.

4. TransitWatch / Buslink World Wide Web Page, *http://www.its.washington.edu/buslink*, downloaded 9/9/98.

5. Busview Plus World Wide Web Page, *http://www.its.washington.edu/busviewplus*, downloaded 9/9/98.

6. Cima, Bart, Smart Trek Deputy Project Manager, Telephone Interview on June 16, 1998.

7. Dailey, Dan, Department of Electrical Engineering, University of Washington, Telephone Interview on June 24, 1998.

8. Dailey, D.J., Haselkorn, M.P., and Meyers, D., *A Structured Approach to Developing Real-Time, Distributed Network Applications for ITS Deployment*, University of Washington.

9. Dailey, D.J., Meyers, D., A *Self Describing Data Transfer Methodology for ITS Applications*, Submitted to IEEE TKDE on 5/23/97.

10. Federal Highway Administration, et. al., *Intelligent Transportation Systems Model Deployment Initiative* [brochure].

11. Goeddel, Dennis, *Benefits Assessment of Advanced Public Transportation System (APTS), Final Report*, U.S. DOT RSPA Report DOT-VNTSC-FTA-96-7, October 1995-July 1996.

12. ITS America, *FAQ - On the MDI, ITS America News*, April 1998, p. 4.

13. Johnson, Christine, and Chen, Kan, ITS Deployment around the World: Lessons Learned, *ITS Quarterly*, Vol. 6, No. 1, Winter 1997-Spring 1998, pp. 23-31.

14. Kirb, Angela, Metro Networks, Inc., Telephone Interview on July 14, 1998

15. Letshaw, Gary, Etak, Inc., Telephone Interview on July 6, 1998.

16. Lundburg, John, Fastline, Telephone Interview on July 13, 1998.

17. Mason, John, ITS - An Opportunity for Local Government to Do Its Job Better, *Institute of Transportation Engineers ITS Council Update*, May 1998, pp. 2-11.

18. Meese, Andrew J., *Transportation Planning Applications of ITS Data*, Presentation at ITS Virginia 4th Smart Travel Conference, Reston, VA, June 1, 1998.

19. Novak, Dave, *Case Study Approach for MMDI Integration Benefits Estimation*, Virginia Tech Center for Transportation Research, Draft 1.5, June 3, 1998.

20. Oak Ridge National Laboratory, ITS Integration Diagram

21. *Operations Working Paper I: Definition of a Classification Scheme for Operations Components*, Draft of 1/26/98.

22. Pacific Rim Resources, *Smart Trek Traveler Information Focus Groups Findings*, 7/24/97.

23. Patton, Brian, City of Seattle?, Telephone Interview on July 1, 1998.

24. Pieratti, Denise, et. al., *Bellevue Smart Traveler and Cellular Telecommunications, Final Report*, U.S. DOT Technology Sharing Program Report DOT-T-93-36, May 1993.

25. Salwin, Arthur E., *Key Findings from the Intelligent Transportation Systems (ITS) Program - What Have We Learned?* Federal Highway Administration ITS JPO Report FHWA-JPO-96-0036, September, 1996.

26. Science Applications International Corporation, *San Antonio Metropolitan Model Deployment Initiative Evaluation Plan*, Draft Version 3.0, March 27, 1998.

27. Science Applications International Corporation, *Seattle Metropolitan Model Deployment Initiative Smart Trek Evaluation Plan*, Version 3.0, March 31, 1998.

28. *Self-Describing Data (SDD) for Dummies*, Version 1.1, September 1997.

29. *Self Describing and Self Extracting Data Flows (SDD/SED) Client User Information Bulletin*, Version 1.0, August 29, 1997.

30. *Smart Trek Stakeholder Interviews: Issues and Opportunities*, Final Report, 12/16/97.

31. *SWIFT Architecture Study*, 6/16/98.

32. *System Engineering Requirements Specification- Smart Trek Metropolitan Model Deployment Initiative Project*, Version 1.0, July 18, 1997.

33. University of Washington ITS Projects World Wide Web Page, *http://www.its.washington.edu/its_projects.html*

34. Watson, Michael S., A New Spin on Bid Sets, *Civil Engineering*, Vol. 68, No. 8, pp. 55-57.

MyBus

10. Transit Vehicle Arrival Prediction: An Algorithm and a Large Scale Implementation [16]

10.1 Introduction

Under the rubric of Advanced Public Transportation Systems (APTS), a number of projects have been implemented to improve distribution of pertinent information (departure time, vehicle delay, vehicle position) about a mass transit system directly to the rider. In this paper, we present an algorithm to predict the arrival of transit vehicles based on a few simple assumptions about the statistics of tracking transit vehicles. The goal of the algorithm presented here is to accurately predict transit vehicle arrival times up to an hour in advance. Beyond the primary goal, there is an additional set of constraints on the algorithm that are imposed to facilitate implementation of the algorithm in real-world systems. These additional constraints are: 1) the uncertainty in the arrival-time is quantifiable, 2) the output of the algorithm is synchronous for display purposes, 3) lost or delayed data is handled efficiently, and 4) the prediction must be statistically better that using schedules alone. In this paper, the time series of time and location pairs is used with historical statistics in an optimal filtering framework to predict future arrivals. We present both an algorithm and a demonstration implementationfor an existing large transit fleet, and compare the results to those for schedules alone.

The prediction algorithm presented here is sufficiently general to be applied to any transit property that (1) operates a fleet which has an automatic vehicle location (AVL) system and (2) has repeated, scheduled service. The demonstration implementation is for the Metro King County transit property in Seattle, Washington.

Optimal filtering techniques are used to develop the prediction algorithm. Such techniques require that the problem be framed in a specific way and that certain properties of the model and model errors be true if the techniques are to be applicable. In this paper, we first examine the underlying assumptions that allow optimal filtering techniques to be used to make the predictions and then detail the construction of the Kalman filter which acts as the predictor.

10.2 Assumptions

Any algorithm has basic underlying assumptions that will color the results. The assumptions used to create this algorithm are:

1. The vehicle locations are available irregularly, typically on a one- to five-minute basis.
2. Each scheduled trip is a realization of the stochastic process of the vehicle traveling the route.
3. The stochastic process is represented by the ensemble of realizations.
4. Vehicles are modeled as if moving with constant speed over a limited distance.
5. Starting and stopping motions of the vehicles are included in the variability of the process model.
6. The variability of the process model is normally distributed.
7. There are known errors in the measurement of the location of the vehicles.

These assumptions allow the problem to be formulated in a statistical framework and fulfill the requirements necessary to use an optimal filtering technique, the Kalman filter, to make optimal estimates of the predicted time until arrival for individual vehicles.

---

[16] Paper presented at the 80th Annual Transportation Research Board Meeting, January 7-11, 2001, Washington, D.C. Authors: D. J. Dailey, S.D. Maclean, F.W. Cathey, and Z.R. Wall.

The first assumption suggests that the time series created by sampling the transit vehicle locations is an under-sampled representation of the actual vehicle path. With such an under-sampled representation, the detailed path of the vehicle cannot be exactly duplicated. This assumption leads us to assert that a detailed vehicle model for the motion is inappropriate for this application and a simple motion model is more appropriate. This model will be the basis for the Kalman filter update equations presented in this paper. The sample rate for the AVL system used in our implementation is on the order of one to three minutes, which is much too slow to capture behaviors such as acceleration or deceleration of the vehicles needed for detailed vehicle models.

The second assumption is based on the observation that the vehicles are operated to meet a regular schedule and that deviations from that schedule represent variability in this process. The bus driver's goal is to meet arrival time constraints that are posted on the schedule; however, each day the driver faces differing traffic conditions that create variability in the arrival time. This suggests that if we observe the vehicle repeatedly on a scheduled daily trip, an average arrival time, with some variability, can be estimated. This notion dovetails with the third assumption.

The third assumption is that we view the observed, scheduled vehicle trips, from different days, as independent ensembles of a process that represents the bus traversing the route. This assumption enables us to use ensemble averaging to obtain overall statistics for the process. A sanity check on the first three assumptions is possible using a time history diagram of vehicle position along a trip. Figure 10-1 is such a diagram, where the position along the trip is on the vertical axis and the time of day is on the horizontal axis. The data points represent the sum of data collected over a period of two months for this route that has six runs at different times of the day. Assumption two is supported by the overall linear shape and even distribution of points for each of the runs, indicating that there is a general trend in the behavior of the vehicle on these trips. Assumption three is also supported by the clear trends and lack of dispersion in the data from a series of days.

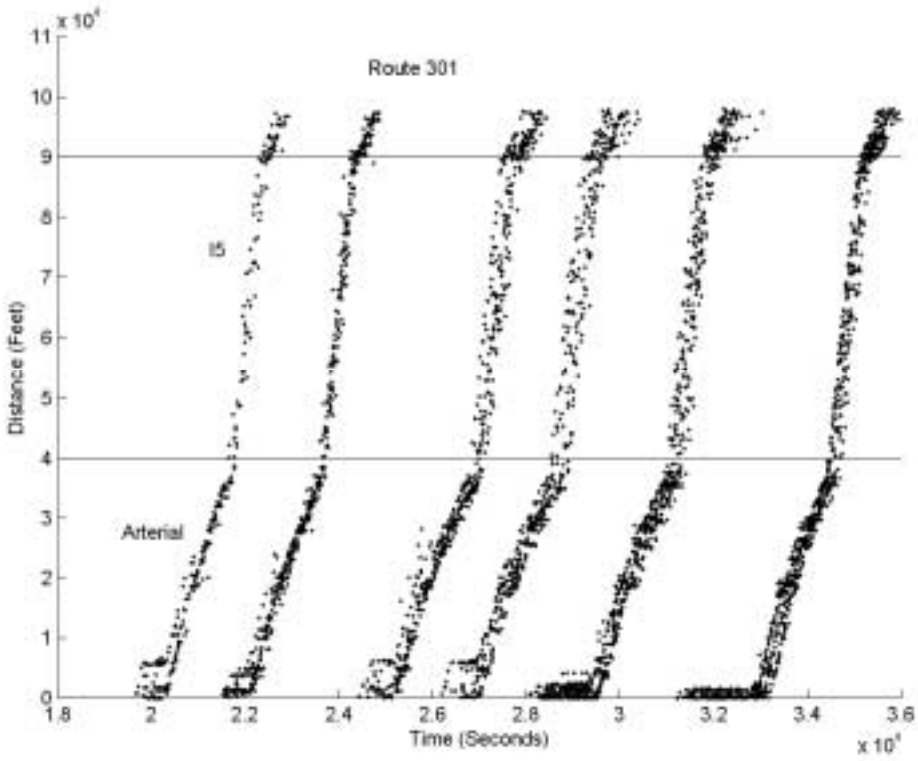


Figure 10-1. Space-time history for multiple ensembles of a set of trips on Route 301

The fourth assumption is that we can create a synthetic "time-to-arrival" function for every destination. This time-to-arrival function $\overline{T}_a(x)$ can be evaluated at each x, such that the time remaining until arrival is

$$\overline{T}_a(x) = \frac{x}{\bar{s}(x)}. \tag{1}$$

The $\bar{s}(x)$ function is analogous to the speed the vehicle would have to travel if it were to travel at a constant speed from location x to the goal. This is not to be confused with the speed of the vehicle at x which is not used in this approach.

It is clear from Figure 10-1 that a globally constant model for $\bar{s}(x)$ is unrealistic for most destinations. Therefore we assert a piecewise constant model. To do this we select a destination of interest and transform the data into pairs of: (1) distance until arrival and (2) time until arrival. We now divide the trip into shorter segments over which we can reasonably assume that the data is approximated as a piecewise constant function in space and time. From this sub-sampling of the trip, we estimate an average overall arrival function value. If the points over the kth interval of size Δx_k about x_k are uniformly distributed over the region, the mean of the temporal values of the points is used to compute arrival time $\overline{T}_a(x_k)$. $\bar{s}_k$ is constructed

$$\bar{s}_k = \frac{x_k}{\overline{T}_a(x_k)}. \tag{2}$$

We define $\bar{s}(x)$ over the kth interval by $\bar{s}(x) = \bar{s}_k$. This approximation is shown pictorially in Figure 10-2, where the observed data is represented by points and $\bar{s}_k$ is the slope of the line between the centroid of the points and the arrival time.

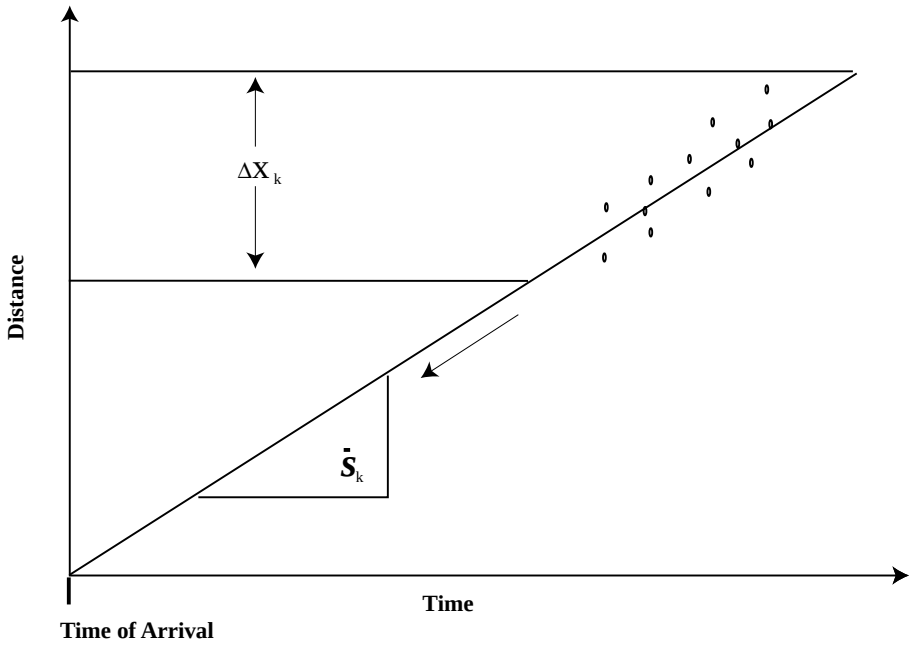


Figure 10-2. Time of arrival function $\bar{s}$

We assume a constant velocity model for subsets of the travel path where the deviations from this model are identified as part of the randomness inherent in the process (e.g., stopping and starting are effectively noise). The deviation of the points from the presumed linear behavior has a probabilistic distribution. To test the properties of this distribution, we fit a line through the points and use the deviations of the individual points from this line as realizations of this probability distribution. We hypothesize that the distribution is Normal, and to test this hypothesis we use a standard distribution membership test, the Kolmogorov-Smirnov (K-S) test.[1] The K-S test computes the probability that two distributions are the same. The K-S test uses a metric of maximum absolute difference between two cumulative distribution functions (D). The metric is used in the computation of the following sum,

$$Q_{\mathrm{ks}}(\lambda) = 2\sum_{j=1}^{\infty}(-1)^{j-1_e-2\,j^2\lambda^2} \tag{3}$$

$$\lambda = \left[\sqrt{N} + 0.12 + .11/\sqrt{N}\right]D, \tag{4}$$

where N is the number of points. Larger K-S test values show that the two cumulative distributions are similar. To apply the K-S test to our data, we use the following steps: (1) Select an appropriate range of samples. (2) Find the sample mean and variance of these points. (3) Generate a normal distribution using the calculated mean and variance. (4) Use the selected points to create a unbiased estimator of actual distribution. (5) Compare the two distributions to find the maximum absolute distance (the K-S Statistic). (6) Use the K-S statistic to compute the probability that the selected points came from the normal distribution.

Figure 10-3 shows the results of this test. It shows the K-S probability for a variety of distances into trips on both the freeway Interstate 5, shown on the left, and an arterial State Route 99, shown on the right. The high probability of distribution membership shown in these figures justifies the fifth assumption of a Normal distribution for the deviation of the observed data from our linear motion model.

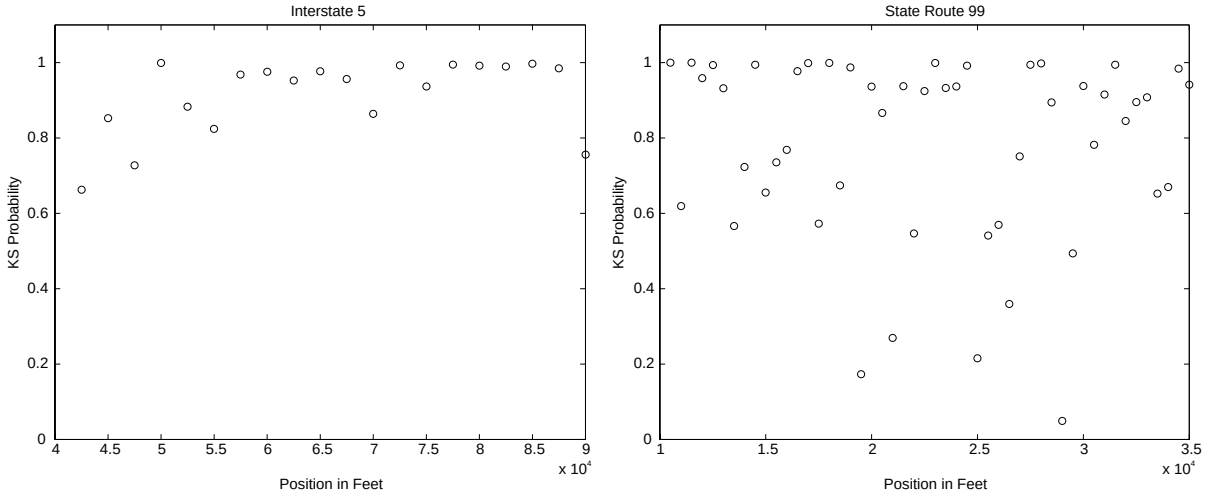


Figure 10-3. KS test results for Interstate 5 and State Route 99

10.3 Algorithm

The prediction algorithm uses a Kalman filter whose state variables at the kth step are the vehicle location x_k, the time t_k, and the time until arrival b_k,

$$\mathbf{X}_k = \begin{bmatrix} x_k \\ t_k \\ b_k \end{bmatrix}, \tag{5}$$

while the observables are the reported position z_k and the reported time of measurement τ_k,

$$\mathbf{Z}_k = \begin{bmatrix} \tau_k \\ z_k \end{bmatrix}. \tag{6}$$

The prediction of time of arrival is done synchronously at each Δt time step, and the observations of vehicle location are reported asynchronously as they are available. The model for the state transition at each time step of duration Δt_k has two parts: (1) a deterministic part involving the state variables, the time step (Δt), and the time-to-arrival function ($\bar{s}(x)$), and (2) a stochastic part (w) assumed to be normally distributed with zero mean. The **time** update equations are

$$\bar{x}_{k+1} = \bar{x}_k + \bar{s}_k \Delta t_k + w_k^x \tag{7}$$

$$\hat{t}_{k+1} = \hat{t}_k + \Delta t_k + w_k^t \quad . \tag{8}$$

$$\bar{b}_{k+1} = \bar{b}_k - \Delta t_k + w_k^b \tag{9}$$

The **data** update for the state variables also have a deterministic and a stochastic component and take the form

$$\hat{x}_k = \bar{x}_k + w_k^x \tag{10}$$

$$\hat{t}_k = \frac{\hat{x}_k}{\bar{s}_k} + w_k^t \tag{11}$$

$$\hat{b}_k = \frac{\hat{x}_k}{\bar{s}_k} + w_k^b . \tag{12}$$

Optimal filtering techniques often assume a relationship between the observables (z_k, τ_k), and the state vector takes the form

$$z_k = \hat{x}_k + v_k^x \tag{13}$$

$$\tau_k = \hat{t}_k + v_k^\tau , \tag{14}$$

where v is the stochastic noise in the observation process. With these variables, we pose the problem in the form of a linear Kalman filter,

$$\mathbf{X}_{k+1} = \mathbf{A}\mathbf{X}_k + \Gamma_k \mathbf{u}_k + \mathbf{w}_k , \tag{15}$$

$$\mathbf{Z}_k = \mathbf{H}_k \mathbf{X}_k + \mathbf{v}_k \tag{16}$$

such that the **time update** matrices are

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \Gamma_k = \Delta t \quad \mathbf{u}_k = \begin{bmatrix} \bar{s}_k \\ 1 \\ -1 \end{bmatrix} \tag{17}$$

and the **data update** matrices are

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \dfrac{1}{s_k} & 0 & 0 \end{bmatrix} \quad \mathbf{H} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}. \tag{18}$$

The generalized solution to the Kalman filter time update is [2]

$$\mathbf{P}_k = \mathbf{A}\mathbf{P}_{k-1}\mathbf{A}^T + \mathbf{Q}_{k-1} \tag{19}$$

$$\mathbf{X}_k = \mathbf{A}\mathbf{X}_{k-1} + \Gamma_k \mathbf{u}_{k-1}, \tag{20}$$

and the data update is

$$\mathbf{P}_{k1} = \mathbf{A}\mathbf{P}_{k-1}\mathbf{A}^T + \mathbf{Q}_{k-1} \tag{21}$$

$$\mathbf{K}_{kg} = \mathbf{P}_{k1}\mathbf{H}^T\left[\left(\mathbf{H}\mathbf{P}_{k1}\mathbf{H}^T\right) + \mathbf{R}\right]^{-1} \tag{22}$$

$$\mathbf{P}_k = \mathbf{P}_{k1} - \mathbf{K}_{kg}\mathbf{H}\mathbf{P}_{k1} \tag{23}$$

$$\mathbf{X}_k = \mathbf{A}\mathbf{X}_{k-1} + \mathbf{K}_{kg}\left[\mathbf{Z}_k - \left(\mathbf{H}\mathbf{A}\mathbf{X}_{k-1}\right)\right]. \tag{24}$$

The covariance matrices for the update and measurement are

$$\mathbf{Q}_k = E\left\{\mathbf{w}_k \mathbf{w}_k^T\right\} \qquad \mathbf{R}_k = E\left\{\mathbf{v}_k \mathbf{v}_k^T\right\}$$

$$\mathbf{Q}_k = \begin{bmatrix} \sigma^2_{w_k^x} & 0 & 0 \\ 0 & \sigma^2_{w^b} & 0 \\ 0 & 0 & \sigma^2_{w^t} \end{bmatrix} \quad \mathbf{R}_k = \begin{bmatrix} \sigma^2_{v^\tau} & 0 \\ 0 & \sigma^2_{v^z} \end{bmatrix}. \tag{25}$$

This algorithm produces the optimal estimate of the arrival time given the information provided to the filter. However, predicting the future is always a challenging activity. To compare the predictions with the vehicle behavior, we need to estimate actual arrival time. Since we are tracking the vehicle irregularly, there is no guarantee that the location will be reported just as the vehicle arrives. To get an estimate of the real arrival, we record the location report just before arrival and just after arrival and linearly interpolate the actual arrival time (T_a). The filter is continuously predicting the arrival as a function of both space and time. We present the

statistics of the deviation of the predictions from the actual as a probability surface in space and time. The left side of Figure 10-4 shows the probability of deviation of the prediction from actual, in minutes, on the front axis (-10 to 10), and the time until arrival for which the prediction was made on the side axis (0 to 30). A similar function for the relationship between the schedule and the actual behavior is shown on the right side of Figure 10-4. The surfaces in Figure 10-4 were created using the predictions made over the course of one day for the second busiest location in the Seattle Metro Transit's service. This location, as well as being the second busiest, is the most complex in terms of the types of trips passing this point. Comparing these two surfaces suggests that prediction with dynamic information has 2 - 4 times smaller errors than using the schedule alone. This demonstrates that there is a significant gain in information over the schedule for transit users.

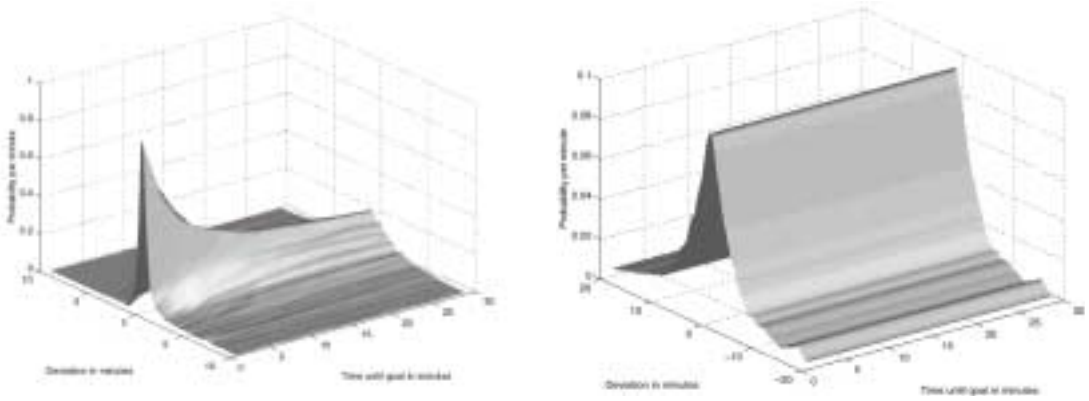


Figure 10-4. Probability of deviation from actual arrival time for the algorithm presented, on the left, and for the schedule, on the right

10.4 Seattle Implementation of MyBus

We have implemented a web-based system that makes predictions for 1200 operating vehicles at each of the 1500 time points identified in the transit carrier's schedule database. We present the implementation by first describing the AVL system used by the transit carrier King County Metro and then detailing the software architecture to support the implementation.

The King County Department of Metropolitan Services (Metro) implemented an Automatic Vehicle Location (AVL) system for its transit fleet in 1992. The AVL system is used by the transit dispatchers for command and control purposes. There are in excess of 1200 AVL-equipped vehicles operating on 240 bus services routes within King County's 2000-square mile service area. Metro's AVL system is composed of three sub-systems: GIS, communication, and data management [3]. These three subsystems work in tandem, and the position information used in the work presented here is obtained by eavesdropping on the communication network connecting the communications and GIS systems. The geo-location of the vehicles is done using odometry and digital maps. Each vehicle has a target and a sensor located on one wheel, and every time the target moves past the sensor, a "click" is recorded. Additionally, when a vehicle passes a signpost transmitter, the on-board receiver records the signpost identification and the time it was passed. The transit vehicles are polled irregularly, but approximately every minute, and the odometry data along with a unique vehicle identification and any signposts encountered, are radioed back to the central management site. The distance traveled along a pre-planned route and schedule information is used in the prediction algorithm as the measurement data.

The algorithm just presented is implemented as a web application called **MyBus**. This application is constructed from a series of collaborating components (see [4]) through which data flow. Each component performs data fusion on the data to add information into the data flow. The component labeled *AVL Predictor* in the lower right of Figure 10-5 implements the prediction algorithm described above. The data stream resulting from this component includes predictions of departures for the vehicles found in the data stream at all of the time points found in the schedule database of the transit carrier.

The component labeled “MyBus.org” in Figure 10-5 is the web server that creates the MyBus HTML screens. An example of this screen is shown at the right in Figure 10-6. The URL http://mybus.org provides a link to the implementation for over a thousand locations in the Seattle Metropolitan region. To demonstrate the portability of the concept we have also implemented the schedule information for Portland and plan in future work to demonstrate real-time prediction for Portland.

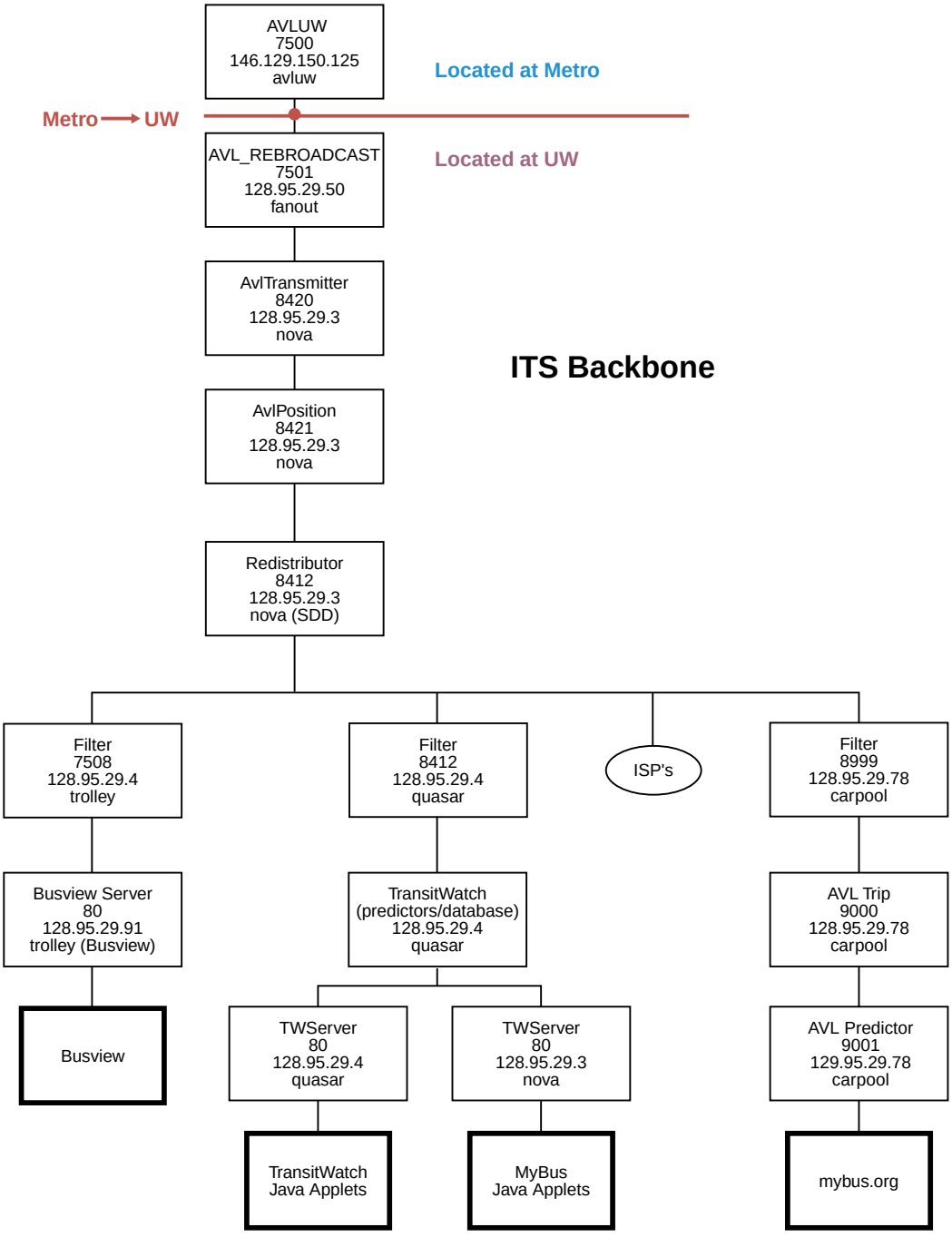


Figure 10-5. MyBus architecture

We conclude that not only is reliable and accurate prediction possible for a large transit fleet, but that there is significant temporal value to the transit-using public in making such predictions widely available.

Figure 10-6. MyBus display

10.5 References

1. W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C*, Cambridge, 2nd edition, 1988.

2. B. D. O. Anderson and J. B. Moore, *Optimal Filtering*, Prentice-Hall, Inc., 1979.

3. D. J. Dailey, M. P. Haselkorn, K. Guiberson, and P. J. Lin, *Automatic Transit Location System. Final Technical Report WA-RD 394.1*, Tech. Rep., Washington State Department of Transportation, February 1996.

4. D. J. Dailey, 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. 163-180, 1996.

11. MyBus: An APTS Based on the US TCIP Standard [17]

11.1 MyBus

The Smart Trek Model Deployment Initiative, a USDOT funded ITS program in the Puget Sound region, has made great strides in integrating and disseminating traveller information. Of particular value is "real-time" information, designed to help travellers make informed decisions about their travel options. As part of the SmartTrek project a variety of real-time transit information applications have been developed. One of these is called MyBus, an application to make departure predictions and deliver traveller information to web browsers.

The goal of MyBus is to present to riders, in real-time, the predicted departure times of buses at specific locations throughout a transit region. King County Metro, the transit agency in Seattle, Washington, USA, operates a large fleet. Up to 1200 vehicles are in service simultaneously, departing from over 1000 locations, called "time points" in the TCIP profile. Predictions are made for approximately 210,000 weekday and 140,000 weekend scheduled departure events, called "trip time points" in the TCIP profile. This is over a million departure predictions per week. The MyBus application has shown that predictions on this scale are feasible and quite manageable.

Several technologies are central to the success of MyBus. A common format for the transit agency schedule and spatial data was crucial, enabling the Seattle pilot project to be re-deployed with the Portland TriMet data with minimal effort. The format chosen was a database schema based directly upon the TCIP standards.

11.1.1 Architecture

MyBus is designed as a distributed application (Figure 11-1). Schedule data and real-time AVL data streams flow between components. These components include a legacy AVL source, a prediction generator, a set of filters for bad or erroneous AVL data, and a web server for final text formatting and delivery over the Web. The extensive use of the ITS Self-Describing Data protocol [1] for inter-component communication ensures reliable and efficient data manipulation and transport. New applications are very simple to build based on reusable collaborative components described in [2]. For example, the Mybus web server is plugged into the output of the Predictor but equally other components could access this data. These downstream components could do such things as on-time performance analysis or congestion estimation.

The standard HTTP/HTML combination is used as the medium for the final prediction information delivery. This assumes that an HTML interpreter is available at the delivery site, which in turn implies some level of computing capabilities at that site. The most common form of delivery is envisioned to be a browser-like screen presentation. However, augmenting this with speech synthesis at the delivery point would allow an audible presentation. Mobile PDAs and cell phones are increasingly Web enabled, and future work on MyBus will accommodate the mobile user.

11.1.2 Prediction

The prediction component of MyBus uses three inputs: (1) a schedule data set, (2) a set of historical trip realisations, and (3) a real-time AVL stream. The schedule data provides the expected time for each event. Historical data provides ensemble-averaged statistics to the algorithm that predicts vehicle departure. The AVL stream supplies instantaneous bus location information approximately every 1 to 3 minutes per vehicle. The latter two inputs are the most important and, in fact, a successful predictor could be built without any temporal schedule information.

---

[17] Paper presented at 7th World Congress on Intelligent Transport Systems, Turin, Italy, November 6-9, 2000. Authors: S.D. Maclean and D.J. Dailey.

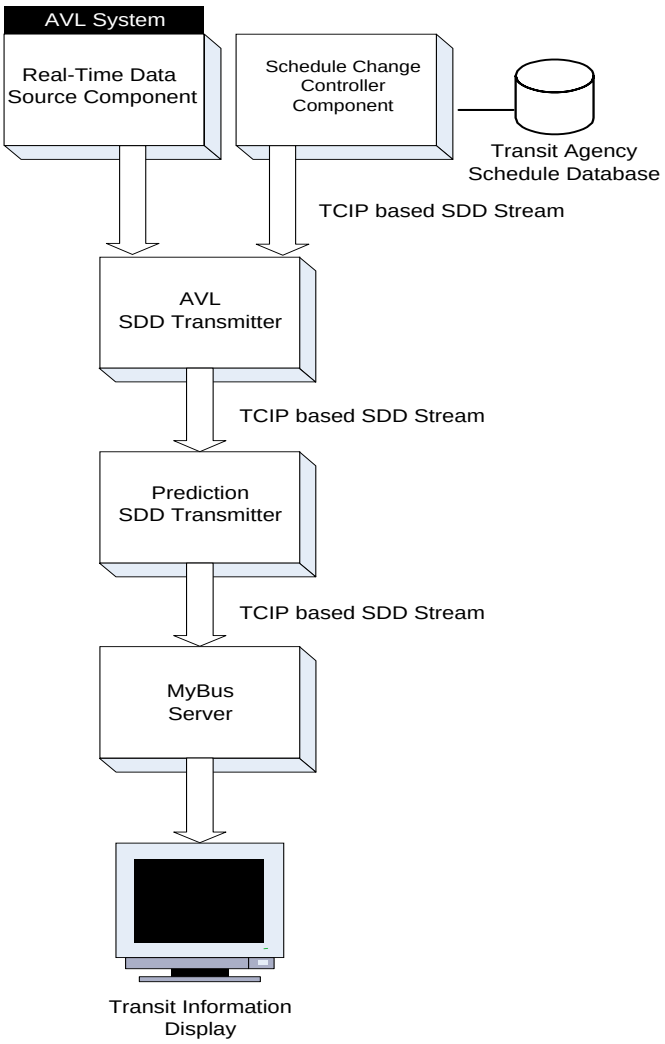


Figure 11-1. MyBus architecture

According to the TCIP standard nomenclature, a bus travels along a block, which comprises a sequence of trips. Each trip is a single run of a bus route in one direction; an example of a trip is the 10:30 a.m., Route 43 bus from Downtown Seattle to the University of Washington. Along each trip, scheduled locations called timepoints exist, generally at major road intersections. These locations are the focus of the MyBus prediction scheme; we designate each scheduled timepoint on the block as a prediction goal. The prediction algorithm tracks each bus to its many goals along the block.

The predictor algorithm [3] uses an optimal filtering technique based on Kalman filter technology [4]. An underlying assumption is that any real-time arrival prediction algorithm depends upon reliable real-time transit vehicle location information. When the maximum number of vehicles are in service (e.g., approaching morning rush hour), the Seattle-based predictor produces up to 25000 predictions every 10 seconds. A separate prediction process is started for each goal to minimise statistical errors. The predictor is started based on either time or distance to goal. For example, the tracking of a bus begins either at a specified time, such as 30 minutes before scheduled departure time, or at a specified distance before the goal, such as 20 miles from the goal. At each time step of 10 seconds, a new prediction is made for both the distance to the goal and the time until departure from the goal. Interleaved with this predictor propagation phase are the real-time AVL inputs, the so-called update phase. The state vector for each Kalman predictor is output at both the propagation and update phases.

In principle, a bus can be tracked from the start of its block to any goal on the block. In practice, the schedule contains out-of-service segments for each block, called deadheads. The current predictor will not predict across a deadhead trip. That is, a goal appearing in a block just after a deadhead trip will not be associated with an active predictor until it is observed via the AVL updates that the bus has completed the deadhead.

The prediction algorithm produces the optimal estimate of the departure time given the information provided to the filter. However, predicting the future is always a challenging activity. To compare the predictions with the vehicle behavior, we need to estimate actual departure time. Since we are tracking the vehicle irregularly, there is no guarantee that the location will be reported just as the vehicle departs. To get an estimate of the real departure, we can record the location report just before arrival and just after the departure and linearly interpolate the actual departure time. The filter is continuously predicting the departure as a function of both space and time. The statistics of the deviation of the predictions from the actual can be expressed as a probability surface in space and time. The left side of Figure 11-2 shows the probability of deviation of the prediction from actual, in minutes, on the front axis (-10 to 10), and the time until departure for which the prediction was made on the side axis (0 to 30). In addition, a similar function for the relationship between the schedule and the actual behavior is shown on the right side of Figure 11-2. The surfaces in Figure 11-2 were created using the predictions made over the course of one day for the second busiest location in Seattle Metro Transit's service. Comparing these two surfaces suggests that prediction with dynamic information have 2 - 4 times smaller errors than using the schedule alone. Moreover, predictions are accurate both far in advance of the scheduled departure time, as well as near departure time. Rider access to this type of information is vital in encouraging mass transit.

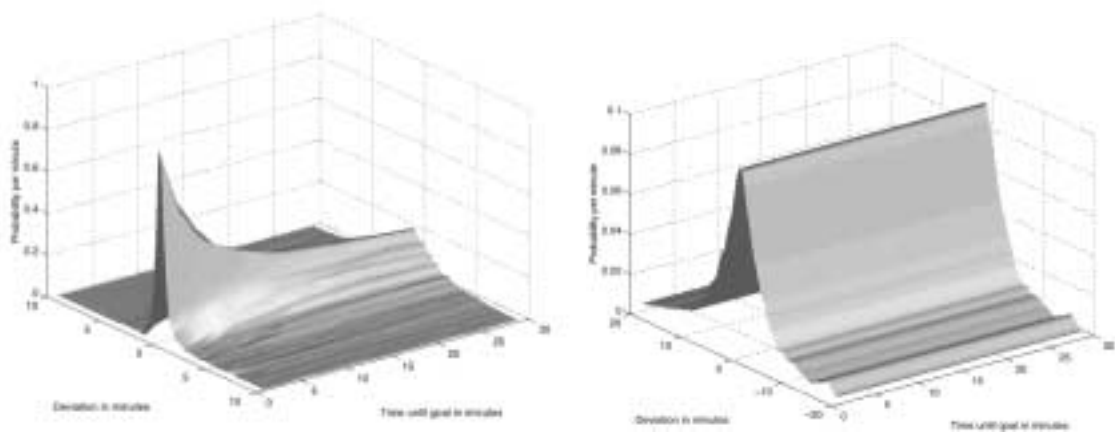


Figure 11-2. Prediction performance

11.1.3 Presentation

The second major component of MyBus is the web server program. This component is responsible for receiving input from the predictor component and for storing and formatting this prediction data in a manner suitable for output over the Internet. An example MyBus display page is shown in Figure 11-3.

Given a prediction, many possibilities exist on how that prediction may be presented. Currently MyBus uses a set of prediction states to determine the message sent to the client. Two departed states exist. The first, bus-observed-to-have-departed-goal, indicates that an AVL update has been received which shows a bus past the goal. The second, bus-predicted-departed, indicates that the Kalman filter has propagated the prediction in the absence of data and is now predicting the bus has departed. Formatted messages directly reflect these different departed states (Figure 11-3, rows 6 and 7).

A filter state showing a positive time to goal is combined with the scheduled time at goal to produces a timeliness report, such as: “Early by 10 minutes,” “On Time,” or “Late by 15 Minutes.” The Kalman filter produces a covariance that is a measure of the prediction’s validity. When a vehicle has not reported its location for a long time, this measure becomes large and eventually the prediction reaches an unacceptable level of significance. In such cases, MyBus switches the display message to “No Information Available.”

The current rule set for mapping predictions into displayed messages is only one of any number imaginable. Due to the distributed nature of the prediction and presentation components, a new display application can easily be substituted. A speech-enhanced MyBus system is under development.

Figure 11-3. Example MyBus web page

11.1.4 Implementation

The MyBus predictor, web server and supporting SDD applications are all built in Java [5]. Currently, the Seattle predictor and the combined Seattle/Portland web site run as two Java Virtual Machines on the same Windows NT host. The ratio of dynamic to static content available from a web site like MyBus is very high. This requires a mechanism for producing the dynamic content and interfacing it with standard web server software, such as Apache [6]. To this end, MyBus uses Java servlet and Java Server Pages technology [7].

The SDD component library used for data transport within the MyBus hierarchy is available at http://www.its.washington.edu/bbone. Prediction information is available under the banner of MyBus at http://www.mybus.org.

11.2 TCIP

The Transit Communications Interface Profiles [8] interface standards for use in the US transit industry. The domain covers the data needs of the functions related to the support of public transportation operations, service and planning[8]. This includes all input and output data for business areas including scheduling and passenger

information, which are of particular interest to the MyBus application. The crux of TCIP is the definition of data elements and how they are presented [8].

From a software architecture standpoint, TCIP is best seen as a benchmark for data structure names, types and relationships. For a rider-directed information system such as MyBus, three standards are of interest. These are the Scheduling/Run-Cutting standard (TCIP 1404), the Spatial Representation (TCIP 1405), and the Passenger Information standard (TCIP 1403) [7]. All are available for review at http://www.ite.org/standards/ntcip/index.htm.

11.2.1 TCIP Integration in MyBus

The schedule information from the target agencies can be represented in terms of TCIP data dictionary data elements with additional relational keys necessary to couple the data elements. Each transit agency is likely to manage its schedule data differently, and so a transformation is necessary for each agency's data into TCIP format. Once the data are placed in this format, they are available to the MyBus prediction and presentation system. To date, the data elements from both King County Metro and Portland TriMet have been successfully mapped into this framework to allow the construction of a TCIP based database.

At the simplest level, TCIP data elements, e.g., timepoint, pattern, trip, etc are mapped into an SQL schema. This schema is called the standard TCIP schema. The tabular structure is used to create the components necessary to support the MyBus application interface. Both spatial and temporal schedule information are necessary to support any predictive algorithm, and both contribute to the schema, e.g., a timepoints table and a trip table both exist. With the inclusion of a real-time AVL stream, real-time predictions on bus departures can be performed.

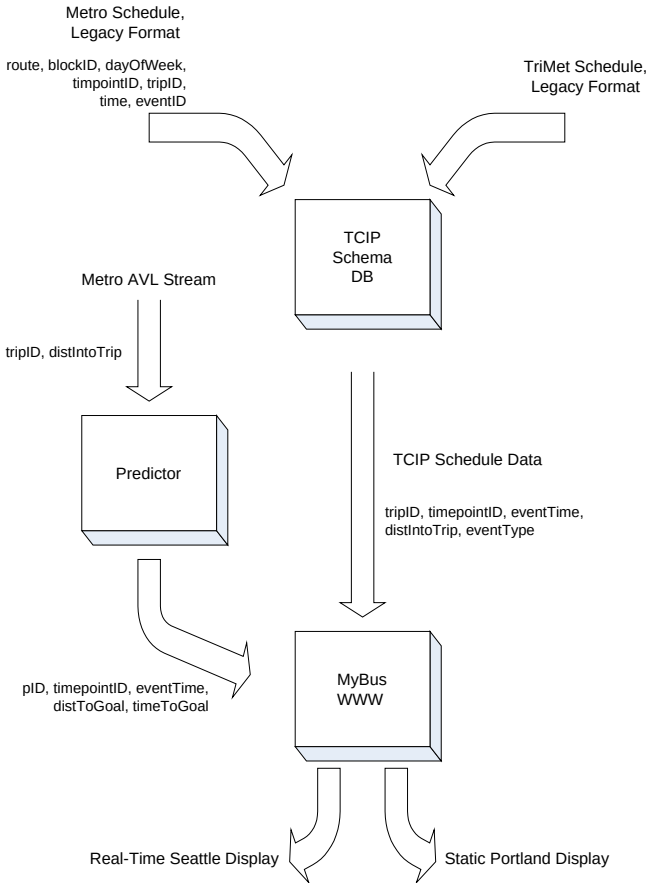


Figure 11-4. Legacy format conversion, static and real-time prediction streams

Furthermore, even without an AVL component, a scheduled or 'timetable' interface can be presented. This system would typically present richer content than available via an agency's own published schedules (Figure 11-4).

11.3 Conclusions

MyBus is an example of combining optimal estimation, distributed computing, information technology, and the World Wide Web to produce an Advance Public Transportation System (APTS) information system. It is designed to support existing web browsers as well as the latest handheld devices. MyBus is framed around the TCIP standard to be portable to any US transit property, and is constructed in a portable language to operate on a variety of computing platforms. It is built modularly to allow for future interfaces. Interfaces presently under design include voice synthesis and recognition systems as well as interfaces to other Advanced Traffic Management Systems where the transit vehicles act as probes to measure traffic congestion.

11.4 References

1. D.J. Dailey, 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.

2. D.J. Dailey, D. Meyers, and N. Friedland, "A Self Describing Data Transfer Methodology for ITS Applications", Transportation Research Record, 1660, pp. 140-147, 1999.

3. D.J. Dailey, Z. Wall, S.D. Maclean and F.W. Cathey, "An Algorithm and Implementation to Predict the Arrival of Transit Vehicles", Proceedings of the IEEE Intelligent Transportation Systems Conference, Dearborn, MI., USA, October 2000.

4. B.D.O Anderson and J.B. Moore, "Optimal Filtering", Prentice-Hall, 1979.

5. K. Arnold and J.D. Gosling, "The Java Programming Language", Addison-Wesley, 1996.

6. Apache Software Project, see http://www.apache.org.

7. J. Hunter, W. Crawford, "Java Servlet Programming", O'Reilly and Associates, 1998.

8. Joint AASHTO/ITE/NEMA Standards Publication, "Transit Communications Interface Profiles (TCIP) Framework Document (NTCIP 1400), Draft 97.01.02", Institute of Transportation Engineers, 1999. See http://www.tcip.org.

12. Web Server Statistics for MyBus.org

12.1 General Summary (MyBus.org)

In this section, we present some statistics obtained from the usage logs created by the Mybus web server. These reflect a snapshot taken on February 15, 2001 for 225 days of operation. At the time of writing, the up-to-date statistics can be found at http://www.mybus.org/stats/www.html. The numbers reported are the total downloads from the pages that display the scheduled and predicted arrival. This page has two components, the textual information about the bus service and one image (GIF) file. So the actual number of pages viewed by users is the numbers from the log divided by 2. The total number of requests over the reported period is 6,004,507 or 3,002,253 actual bus status pages viewed.

The Figure 12-1 shows the increasing trend for the use of Mybus.org with over 1.6 million requests for Mybus pages in January 2001 alone.

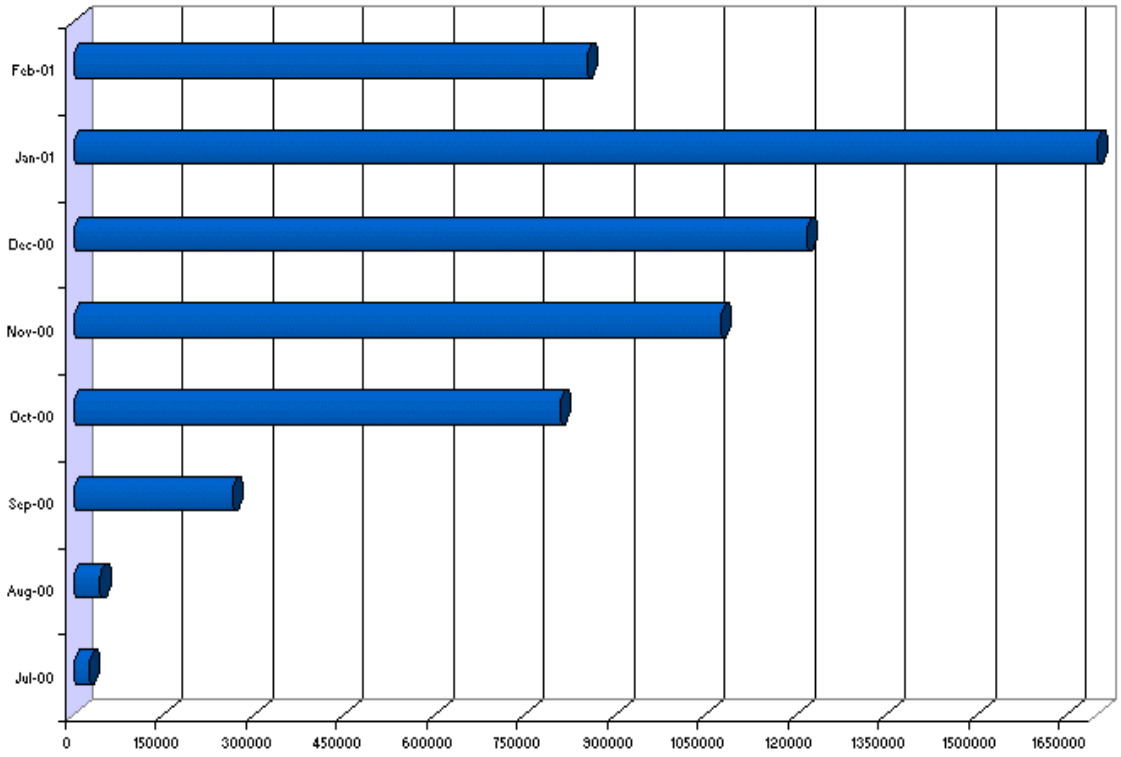


Figure 12-1. MyBus monthly report

Figure 12-2 shows the use by day of the week. The greatest usage occurs during the Monday through Friday period. The speculation is that the application is being used to assist in commuting behavior.

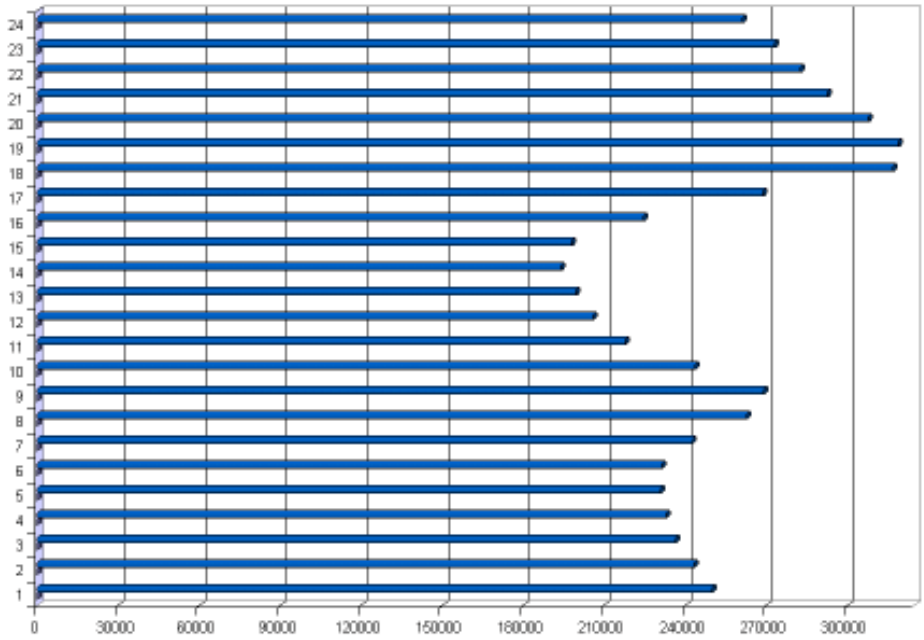


Figure 12-2. MyBus daily summary

Figure 12-3 shows the usage by time of day. There are peaks during the morning and evening commute with the evening peak being larger. This leads to the suggestion that the application is most frequently used by bus riders at work to plan the evening commute.

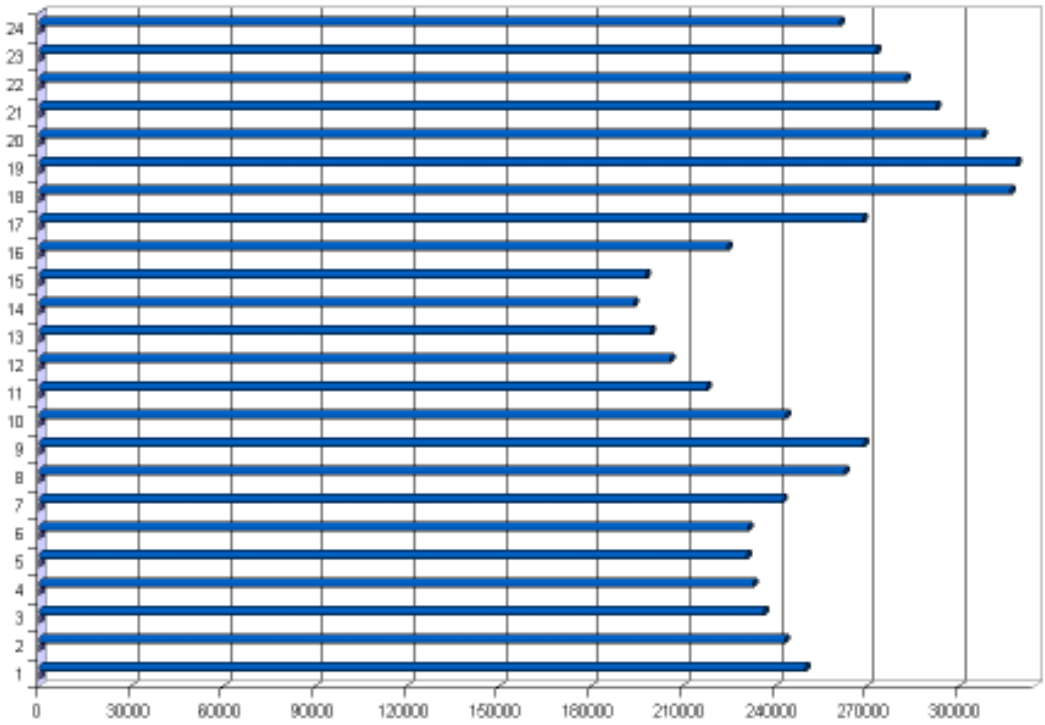


Figure 12-3. MyBus hourly summary

Figure 12-4 shows that the biggest users are large local employers, such as Microsoft, the University of Washington, and Amazon.com, as well as Internet service providers like AT&T through @home.com and USWest.

Requests	Organization
426,520	microsoft.com
847,819	[unresolved numerical addresses]
729,301	washington.edu
333,810	amazon.com
299,128	home.com
251,541	oz.net
234,626	uswest.net
219,120	prognet.com
195,872	eskimo.com
134,366	level3.net
96,962	f5.com
94,810	primus.com
83,762	gtei.net
77,470	nwlink.com
54,657	ci.seattle.wa.us
48,446	dsl-isp.net
46,952	aliaswavefront.com
46,161	speakeasy.net
33,127	riochet.net
30,508	viair.com
30,071	avenuea.com
29,983	att.net
28,672	esca.com
28,425	popsite.net
25,533	accessone.com
586,865	[not listed: 1,671 organizations]

Figure 12-4. Requests by organization

Requests	Browser
4,255,615	MSIE
1,506,603	Netscape
195,990	Lynx
4,719	Netscape (compatible)
3,691	Snotzilla
3,356	lwp-trivial
3,135	PHP
1,740	MSProxy
1,081	WebTV
775	I am not a user agent; I am a free man!
642	MARS VC
466	Python-urllib
455	Googlebot
367	Opera
254	Slurp
178	Microsoft URL Control - 6.00.8169
173	curl
159	UP.Browser
104	Active Cache Request
98	SpaceBison
1,604	[not listed: 185 browsers]

Figure 12-5 shows the type of browsers used to interact with Mybus. The majority of the requests come from either Microsoft Internet Explorer or Netscape Navigator.

Figure 12-5. Requests by browser

12.2 General Summary (MyBus.org–WAP)

In this section, we present some statistics obtained from the usage logs created by the web server that deploys Mybus for WAP phones. These reflect a snapshot taken on February 15, 2001 for 140 days of operation. At the time of writing, the up-to-date statistics can be found at http://www.mybus.org/stats/wap.html. The numbers reported are the total downloads from the pages that display the scheduled and predicted arrival. There where 10,994 total pages requested from the WAP phone service.

The Figure 12-6 shows the increasing trend for the use of Mybus.org via WAP phone.

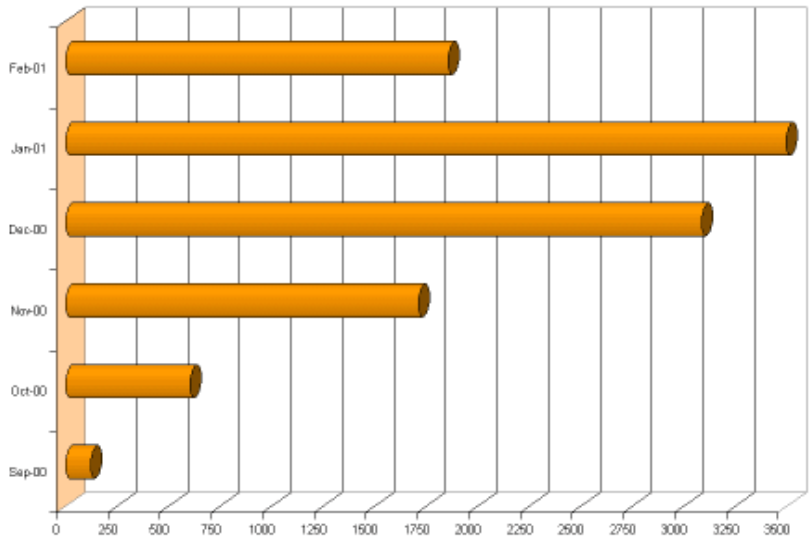


Figure 12-6. WAP MyBus monthly report

Figure 12-7 shows the use by day of the week. Most usage occurs is during the Monday through Friday period. The speculation is that the application is being used to assist in commuting behavior.

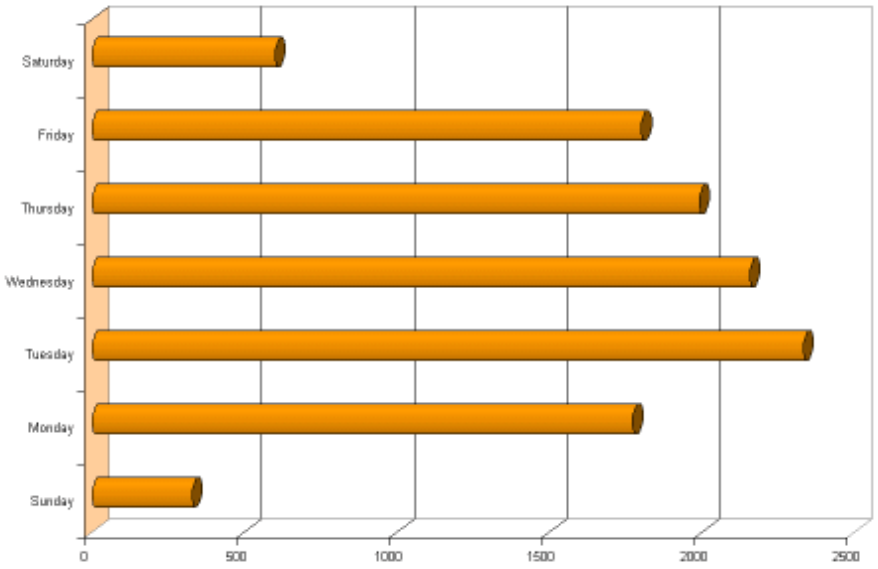


Figure 12-7. WAP MyBus monthly report

Figure 12-8 shows the usage by time of day. There are peaks during the morning and evening commute, with the evening peak being larger. This leads to the suggestion that bus riders most frequently use the application at work to plan their evening commute.

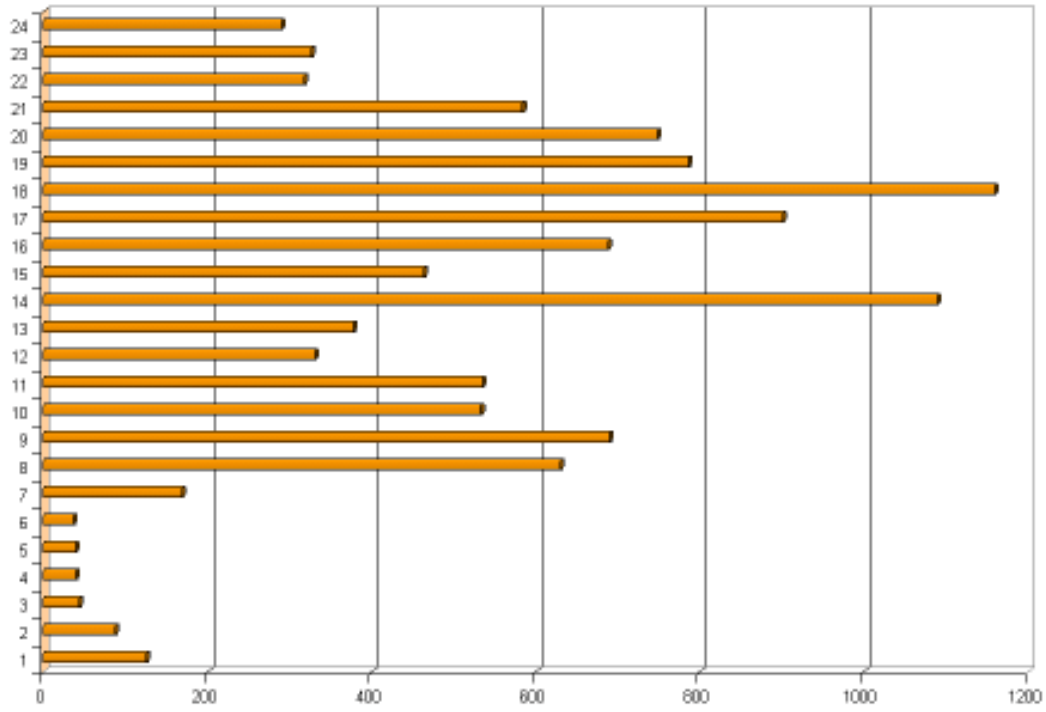


Figure 12-8. WAP MyBus hourly summary

Figure 12-9 shows that the largest users are wireless service providers.

Requests	Organization
7,334	[unresolved numerical addresses]
1,909	airdata.com
707	uswest.net
127	washington.edu
90	materna-com.de
86	ministart.com
77	microsoft.com
71	phone.com
71	avenuea.com
54	home.com
36	eplus-online.de
30	viair.com
27	voicestream.com
21	mindspring.com
21	telinco.net
15	thewebzone.net
14	oz.net
12	t-dialin.net
11	surf-callino.de
11	cheapnet.co.uk
11	pinpoint.com
10	wapsilon.com
10	uswest.com
9	interliant.com
8	urscorp.com
222	[not listed: 115 organizations]

Figure 12-9. WAP requests by organization

Figure 12-10 shows the type of browsers used to interact with Mybus WAP. The majority of the requests come from UP.Browser.

Requests	Browser
9,702	UP.Browser
339	MSIE
129	Netscape
124	WAPman Version 1.7.2:Build P2000112400
90	Materna-WAPPreview
86	4thpass.com KBrowser 1.0
66	UPG1 UP
51	YourWap Nokia7110
37	Nokia7110 (DeckIt
36	Nokia6210
26	Wapalizer
25	MOT-CB
25	WAPman Version 1.7.1:Build W2000092200
22	WAPman Version 1.7.1:Build P2000092200
18	Nokia-WAP-Toolkit
17	SAMSUNG-SGH-Q100
16	Opera
14	M3GATE
13	YourWap Motorola 7389
12	Klondike
143	[not listed: 29 browsers]

Figure 12-10. Requests by browser