Center for Urban Studies Publications and
Reports

Center for Urban Studies

3-1998

Transit Time Internet Access

Janet Vorvick
Portland State University

Kenneth Dueker
Portland State University

Transit Time Internet Access

Janet Vorvick
Center for Urban Studies
Portland State University
Portland, OR 97207-0751
503/725-4020
503/725-8480 fax
janetv@iname.com

Kenneth J. Dueker
Center for Urban Studies
Portland State University
Portland, OR 97207-0751
503/725-4020
503/725-8480 fax
duekerk@pdx.edu

ABSTRACT

Transit Time Internet Access (TTIA) is a World Wide Web (WWW) application which delivers real-time bus schedule information to users of the Internet. TTIA allows a bus rider to request and receive schedule deviation information about a specific bus at a specific timepoint. This report explains the design of TTIA, the specifics of the implementation, the issues of scope, the problems that were encountered and some conclusions.

TTIA is accessible at http://www.upa.pdx.edu/TTIA/

INTRODUCTION

Transit Time Internet Access (TTIA) is a prototype of a Web application that allows bus riders to receive information about a bus they intend to ride. Through the TTIA Web pages, a user of the bus system of Tri-Met, the Tri-County Metropolitan Transportation District of Oregon, can find out if their bus is expected to come to their bus stop early, late or on schedule. TTIA is an exploratory research project to examine innovative ways of delivering bus schedules and actual arrival times to customers. The ability to deliver actual arrival time information has been made possible by Tri-Met's implementation of a new bus dispatch system (BDS). The BDS package includes automatic vehicle location (AVL) technology that uses a global positioning system (GPS) to track the location of buses. Bus locations are then related to digital representations of bus routes and schedules.

Currently, users can access schedule information by phone or on the Internet. Tri-Met is installing a new telephone information system that will deliver actual arrival time information as well as schedule information. In fact, the TTIA system taps into the stream of exception messages sent by the BDS for the new telephone system, and processes it for delivery via the Internet.

TTIA is part of a larger project evaluating uses of data coming from the new BDS. This larger purpose addresses the implications of improvements in the quality of transit service. Examined in this context is the manner in which better arrival time information will change users' travel behavior. Will it increase transit patronage or reduce the erosion of ridership over time? Similarly, can analysis of on-time performance and bus bunching lead to strategies and procedures for dispatcher interventions that can improve on-time performance?

A user of TTIA receives an actual arrival-time estimate by opening a Hypertext Markup Language (html) document (html encodes the instructions to a Web browser that allow Web pages to be displayed with links to other pages). This corresponds to using a browser to click on a link on a Web page. Assisting the user, TTIA offers a series of four choices that determine which bus is the user's bus-of-interest and at what location they plan to board. The schedule update page for the user's bus-of-interest reports either that the bus is early, is late, is running on time or that there

is no information available for the bus-of-interest. Proposed extensions of TTIA would report reroute information, (snow routes, for example) and also alert the user if the bus-of-interest is out of operation.

TTIA is intended for the use of regular bus riders. Therefore, some aspects of the bus system don't need to be explained to the user as he/she is guided through the Web pages that offer a choice of routes, etc. The assumption is made that TTIA users know where they are, know where they want to go, can read Tri-Met's printed bus schedules and can navigate the Web. Users who haven't experienced the Web have to learn only the actions that any Web user learns: point and click to open a page and bookmark those pages used often.

The top page of TTIA offers a choice of routes. The next page offers a choice of directions which usually correspond to inbound and outbound. The third choice is the choice of a bus stop. The choices offered correspond to the information provided on a paper schedule – ten or twelve timepoints along the route are listed – and help is offered in the form of maps or more detailed stop lists. After selecting a location, the user is presented with a list of scheduled bus arrival times at that location for a period of time that begins fifteen minutes before the current time and ends 45 minutes after the current time. The user then clicks on the arrival time of the bus they plan to take causing the execution of a program that searches a database of messages to discern whether there is a reported deviation for that particular bus. The information from the appropriate message is relayed to the bus rider in a Web page.

One of the greatest challenges for TTIA is the interpretation of the messages in the database. The messages used by TTIA are schedule exception messages which are automatically sent by radio to the BDS from a bus which is off schedule. The absence of a message for a certain bus usually means that the bus is on schedule. However, it can have other meanings, as will be discussed later. A message indicating that the bus was reported to be late may mean that the bus will be late to the user's stop, or it may not, since the bus may make up lost time. The user is presented with a disclaimer explaining the limitations of the system so that they understand what the information means – and what it doesn't mean.

TTIA is a prototype rather than a commercial system. The construction of a commercial system precludes the flexibility and speed desired for the development process. Specifically, the development process which was employed is called *rapid prototyping*. Rapid prototyping is a technique that addresses the problem often encountered at the end of software projects that the final product is a program that no one wants. Eliciting information from the end user about the program they want is so difficult that discussion has come to be seen as inadequate. To facilitate the expression of the requirements for a program, a prototype is created and shown to the user. However, no attempt is made to give the prototype full functionality.

In addition to the absence of full functionality, a program that is part of the rapid prototyping process of software engineering often lacks the organization and structure of a production program. The purpose of the prototype is to facilitate the specification of the program to be made, so it isn't important to make it tidy. Even the choice of languages and software tools can be different for the prototype and the commercial system. It's not unusual for the system designer's idea of the program and the user's to be substantially different even after the initial discussions of the problem to be solved. So it would be surprising if the programmer happened to choose the structure and language for the prototype that turned out to be the best for the commercial system. The implication is that one does not improve the prototype to create the commercial system. One discards the prototype and begins the implementation taking advantage of all the information learned in the rapid prototyping process.

As a consequence of the use of rapid prototyping, it was not desirable to incorporate every Tri-Met bus route into the TTIA system. To reduce the scope of the project, the weekday schedules for seven routes were used. Also, because of the limitations of a small scale prototype, some attractive features of TTIA were only partly implemented. For example, a program to help a bus rider locate their stop if it is not a timepoint is not complete, and the exception messages TTIA delivers are drawn from a static database of schedule exception messages. Acquiring the schedule exception messages in real-time was hampered because communications between TTIA and the Tri-Met computer receiving the exception reports have been problematic. Still, the prototype is

adequate for assessing the usefulness of a real-time information delivery system. The experience gained from creating the system allows the suggestion of some design and implementation pitfalls to avoid.

LITERATURE REVIEW

TTIA is designed to deliver real-time information to a transit user. Similar systems are in place serving the riders of several transit systems in the U.S. A larger number are planned or under development. The following section looks at some of the real-time systems for transit users.

Systems that deliver real-time information are generally classified as pre-trip or en-route traveler information systems. TTIA is designed to be a pre-trip information system, but differs from most pre-trip systems in its delivery of real-time information. Most pre-trip systems are designed to help a transit user plan a trip. Delivery of the information one would find on a printed schedule is almost universal, but a real-time component is rare.

Among the pre-trip systems that, like TTIA, provide real-time information is the Busview(1)(2), a Web application developed by the University of Washington and the King County Department of Metropolitan Services. This program shows, on a map, the location of busses on a U.S. TIGER census file representation of the street network. Like TTIA, Busview allows users to plan their walk to the bus stop based on real-time information in addition to fixed-schedule information. Busview uses a signpost-based AVL system to locate busses. Users query a bus icon to obtain schedule information, but must personally interpolate from the current location of the bus to estimate an arrival time at their bus stop.

Another pre-trip system for delivering real-time information to bus riders is operating in Minneapolis. Two methods of delivering the information are in use by this system called Travlink(3)(4). First, Videotext terminals have been distributed to a group of people recruited for the project. Data collected indicate that this method of accessing real-time bus information is attractive to users. Second, kiosks provide real-time information. At the three kiosks, real-time information is not used much. The kiosks are used mostly for trip planning.

Several other pre-trip systems were in the planning stages as of January 1996. Kiosks that offer trip planning augmented with real-time information are part of the transit plans in Corpus Christi(3), Cincinnati(3)(5), and Atlanta(3)(6). In Ann Arbor, Michigan, plans include a demonstration of wayside signs that give the amount of time until the next bus's arrival. Cable TV is also among the media being considered in Ann Arbor(3).

TTIA is similar to en-route systems and could even become an en-route system if a computer in a kiosk were located at a bus stop. TTIA has a lot in common with the twenty-one en-route systems in North America identified in the U. S. Department of Transportation's 1995 report, <u>Review and Assessment of En-Route Transit Information Systems</u>(7). A few of these use on-board signs to display real-time information and are, therefore, very different from TTIA. But most use kiosks.

TTIA DESIGN

There are basically two parts of the TTIA system. One part collects information from the user and the other delivers information to the user. Both of these parts use a Web browser as their user interface.

Collecting information from the user involves some data about the bus system. For example, the user chooses among the bus routes as they indicate the bus in which they are interested. Thus the information collecting part of TTIA presents the user with choices and records the choice. There are four choices to be made:

- What route?

- Direction of travel? Inbound or outbound?

- Which time point location?

- What time?

For example, if the answers were route 8, to Portland, SW 6th and Main and 10:15, it would mean that the user plans to board bus 8 at SW 6th and Main at 10:15 and wants to know if it's running late.

A design decision had to be made between creating on-the-fly Web pages to offer the appropriate choices as the user proceeds and creating all of the Web pages ahead of time and simply choosing the right page to serve as the user proceeds. The first approach is dynamic in the sense that pages are made only as they are needed. The second approach is static because the pages are created just once and then they sit around until they are needed.

Since the price of storage space on a computer is small, creating all the pages just once is reasonable. But TTIA with its seven bus routes requires 600 Web pages to offer all the options of route, direction, stop and time. Creating Web pages on-the-fly reduces the amount of space needed, but requires calculations every time a user clicks on a page. The design decision for TTIA was in favor of static pages partly because static pages were already being created by Tri-Met to provide bus schedules on the Web.

The final step in the information-gathering part of TTIA occurs when the user clicks on a time (the scheduled time the bus comes by their stop). The Common Gateway Interface-binaries (cgi-bin) facility of the Web server (the machine on which the Web pages are located) allows the four pieces of information to be communicated to the second part of TTIA, the part which delivers the schedule adherence information. The major work of the second part is the creation of the update page for the user's bus-of-interest.

To create the update page, a program is executed. It receives the user's input and checks the database of schedule exception messages to see if the bus-of-interest is currently known to be off schedule. This database of exception messages is indexed by an identifying number for each bus. But the identifiers are not encodings of the route, direction, and time, so it is necessary to look up the identifier for the bus of interest. For example, the user who plans to board bus 8 headed to Portland at SW 6th and Main at 10:15 wants a specific bus running a trip the Tri-Met refers to as Train 1733. If the database contains an exception message for Train 1733 which has in the *time* field the number 11, then the bus is eleven minutes late.

Naturally, the details of the database are crucial. The design specification for the update page generation need only assert that the database is consulted and the required information is obtained.

Using the cgi-bin facility again, the output of the program is sent to the user as a Web page. Most of the update page is the same for every inquiry. The parts that change are the estimated time that the bus will arrive at the user's stop, the repetition of the information identifying the bus-of-interest (for the user's benefit) and the disclaimer that accompanies an estimated arrival time.

A picture of the flow of information through the TTIA system can be seen in Figure 1.

TTIA: A MANAGEABLE SIZE

The production of a commercial system for delivering real-time schedule adherence information is no small task. Even the production of a prototype of such a system, such as TTIA, is a major undertaking. Reducing the task to a manageable size was one of the first and most significant challenges in the development of TTIA. Two strategies were employed. First, some features were implemented fully, but enabled for only some part of Tri-Met's system. Second, some attractive features of TTIA were delayed while the most fundamental features were implemented.

The implementation of TTIA for only seven of Tri-Met's one hundred routes and only for weekdays reduced the number of static Web pages needed from 4500 to 160. Even 160 Web pages cannot be generated by hand, especially when changes need to be made frequently. The TTIA pages were generated by a perl (a language used for easily manipulating text, files and processes) program. When changes were made to TTIA, the perl program was changed and the pages regenerated. This process would have taken much more time for the full system. Also, the increased number of routes with some anomalous feature that required special attention would have been problematic.

To reduce the scope of the project, TTIA was first implemented without maps, without the module for helping riders find their bus stop (if it is not a time point) and without scrolling boxes or frames. This simpler version of TTIA was completely adequate for demonstration to potential users and other interested persons. The comments which were received about the early TTIA guided the choice of features to add.

Some of the features that were not added to TTIA would be very useful for a commercial TTIA-like system. Under the heading of "rider training," for example, a component for helping users learn to read bus schedules would have been good to include. The assumption that the users of TTIA are already bus riders and that they know which bus they want to catch precluded the need for writing any trip planning pages or programs. But a commercial system wouldn't ignore the new bus rider, the out-of-town visitor, the rider headed for some unfamiliar destination or the rider for whom schedules are indecipherable. Even sophisticated riders may need some clues about the TTIA information to avoid making mistakes. For example, if a rider uses TTIA and learns that his or her 10:05 bus is ten minutes late, he or she may decide to take the 10:25 bus instead. TTIA doesn't warn the user if the two busses don't serve all the same bus stops beyond the user's place of boarding. The user is responsible for knowing exactly which busses are suitable. Thus an important recommendation is the inclusion of the overhead sign designation that provides users information about the bus destination.

Many other aids to the user were attractive but beyond the scope of the TTIA project. For example, TTIA requires the user to specify their route before specifying their direction of travel and bus stop. It was beyond the scope of the project to structure the pages in a way that would allow the user to indicate that he or she is interested in any bus that will travel inbound, for example, from a certain stop. A related modification would allow the user to state the information concerning route, direction and bus stop and ask when the next bus will arrive. For the user unfamiliar with the area they're in, TTIA could offer help locating a nearby bus stop. TTIA allows the user to click on a time point of a schematic route map, which results in a display of the schedule at that time point. Thus the user does not need to know the timepoint name. An improvement would be to allow users to click anywhere along the bus route, bringing up a list of stops. Better yet, smart maps would produce some result when the user clicked anywhere — even off the bus route. The map could zoom in from the schematic route map and display the selected area with a detailed street map showing individual bus stops as well as time points.

If a TTIA user wants to know each day whether a certain bus is on schedule, say the bus that they catch home from work, some kind of reminder could be communicated to them. Though this was originally within the scope of the TTIA project, it was not implemented because such a service would require some kind of subscription on the part of the user and, possibly, the distribution of software. A Web browser is a standard piece of software and the assumption that users have a browser doesn't limit TTIA.

Laying aside these particular ideas, the scope of TTIA was reasonable. Set aside were additional cosmetic improvements. Focus was on functionality and user-friendliness.

ISSUES AND OBSTACLES

As TTIA was implemented, many unexpected problems arose. In addition, issues that had been considered briefly came up again as decisions had to be made. Quite a few of these issues and obstacles will rear their heads in any real-time information delivery project. This section introduces and discusses those that are most important.

The first attempt at a user interface adopted the structure of the Tri-Met schedule pages from the Web. Tri- Met displays a grid of times for which the rows are labeled with the timepoint names and the columns (though not labeled) correspond to distinct trips, each with its own Train number. The users who saw these pages found the information too dense. The choosing of a time point and a bus arrival time in one step required too much information to be displayed on a single page. The solution was to break the choice of a time point and the choice of a bus arrival time into two steps. However, this greatly increased the number of static Web pages needed.

Not only was the structure of the Tri-Met schedule pages adopted, it was used as input to a perl script that generated the early TTIA pages. This had an unexpected effect. Since the Tri-Met pages are displaying the bus arrival times for use by a human, the individual times on the screen are not hot links — that is, they are not clickable. But the bus arrival times displayed by TTIA are hot links, and the information generated is used by a program. It became apparent that the kind of information required had been omitted from the Tri-Met pages because riders don't need it.

Specifically missing were the Train number for an individual bus arrival time and the location identifier for a bus stop. Had the Tri-Met database been the starting point, the program could have generated Web pages that included the Train number and location identifier very easily. In fact, the program used at Tri-Met to generate the schedule pages for the Web *does* use the output of a database query as its input. To use the Web pages as input to TTIA's Web page generating program was good for rapid prototyping, but not good in the long term.

Some of the peculiarities of the kind of data that represent a transit system cropped up as obstacles for TTIA implementation. A greater familiarity with the data that represent a bus system might have forestalled some trouble. For example, some bus stops are visited twice on a route. Working on the assumption that any bus stop is visited only once by any bus as it travels in one of the two directions on the route, the TTIA Web page generating code was written to place in the list of the times that a certain route serves a stop one entry for every Train number. However, more than one bus in the small sample looped, visiting a stop twice for a single Train number, as shown in Figure 2.

The code should have assembled the list of times the bus passes stop B by collecting and sorting the *times* B is served by that route, not the Train numbers that serve B. The list of times must reflect that the next bus arriving at stop B may have the same Train number as the previous one (and is, in that case, the same bus).

In a similar vein, the significance of the fact that some bus routes follow different paths at different times of the day was overlooked initially. For example, one bus may serve all the stops along a route from the city center to a distant suburb, but the next bus, which has the same route name and number, may end its service half-way along the route. Express busses produce the same effect. Some stops are part of the route, but the express bus doesn't stop at them. These anomalies have stranger companions. Figure 3 shows that some routes have alternate stops, so that one bus takes the path A, B, C, D, E, I and the next bus serving that route takes path A, B, F, G, H, E, I.

These situations pose a problem for TTIA when the user asks for help locating their bus stop. Which list of stops shall be given them to examine? If they are given all the stops, will users assume the order in which the stops appear in the list is the order in which they are served? It was decided to allow users to examine the list of stops along the path served by the next bus (with respect to the time of day that they are making their request to TTIA). If they don't find their bus stop, they can request another list of stops to examine.

Enabling users to choose a bus stop between time points leads to a calculation of the time the bus will reach that bus stop based on the time the bus is due at the nearby time points. The Interpolated Time Module (ITM) calculates the arrival time of a bus at a non-time-point stop, but routes that have several paths can complicate this. Consider a case in which I, D and H are time points, but E is not. If a user chooses E as their bus stop, which are the nearby time points, I and D? Or I and H? This is another example of the sometimes subtle challenges that arose as the implementation progressed.

To return to the decision to use static Web pages, there were several issues of interest. First, the 160 static pages have to be regenerated every time the bus schedules change. Major changes occur four times a year. This means that someone must be in charge of the maintenance of any TTIA-like system, unless the pages are not static. If the pages are created dynamically from a database of route information, the changes in the data base will be enough to update the real-time system.

Second, the tree structure of the TTIA pages made it imperative that a user provide the information needed to deliver the message about their bus in the order TTIA assumes. If a user wanted to indicate their stop first, the TTIA pages would not prompt them for the remaining information. A more sophisticated algorithm for collecting information would improve TTIA whether the pages were static or not. If the pages were created dynamically from a database, however, it would forestall the explosion of static pages needed to allow the user to provide information in a different order than that which TTIA uses.

Third, it was desirable to add a dynamic component to the fourth level of static Web pages. For a bus that runs frequently during the day, the number of times that a fourth level page needs to display to the user is large — more than a hundred. To make the page more readable, it was assumed that the user was interested in update information about a bus that is arriving at his or her stop soon after the time that the request is made. So the user's action, opening a URL, causes the execution of a perl program that filters the static page with its many times. The page is displayed without the times that are more than 15 minutes in the past or 45 minutes in the future. Thus, the page is static but the page is edited dynamically.

On the topic of the real-time information itself, there were some significant obstacles. First, it was necessary to decide how many exception messages to save for use by TTIA. Since another application at Tri-Met uses the exception message, there had been some planning for the transmission of exception messages on a local area network. Saving only one exception message for each Train number would allow a report that a bus was running late, but would not allow any deducing as to the bus's situation. For example, an exception message that says that Train 1703 is 8 minutes late gives no clue about the bus's situation. But a bus reporting 8 minutes late which reports it is 10 minutes late and then 12 minutes late at 2 minute intervals is a bus that is not moving at all. It was decided that deducing information based on the exception messages was beyond the scope of the project and thus, TTIA saves just one exception message for each Train number.

The other issue concerning exception messages was the delays that have occurred in the installation of Tri- Met's BDS. Exception messages are collected primarily for the use of dispatchers who are monitoring and adjusting bus service moment to moment. The delivery of exception messages to TTIA had not been arranged as of the final prototype. The availability of several groups of test data made it possible to create a reasonable database of exception messages, but real-time data is not available.

Update pages were another cause of serious consideration. By showing TTIA to potential users, it became apparent that technical language had to be avoided in the user interface. For

example, at first TTIA included the phrase "estimated arrival time," but some users found "arrival"

confusing since they were departing on their trip. The format settled upon doesn't use the

estimated arrival time in a sentence. It is just displayed on the screen under the heading "Update

Information."

   Update pages were an issue with regard to disclaimers, too. When a bus is reported late, the

user needs to be alerted that the bus may not continue to run late. The bus could be on time or

even early to their stop. One problematic situation involves layovers at the end of a trip. For

example, if a bus is twelve minutes late as it nears the end of a trip but is scheduled for a 15 minute

layover to give the driver a break, the bus may very well be on time as it starts the next trip. TTIA

doesn't attempt to identify these situations, but delivers a disclaimer that may help users consider

that possibility.

   Another disclaimer is needed because an unsophisticated user might request information about

their bus 15 minutes before the bus is scheduled to arrive and not ask again. Since a browser will

continue to display the last Web page accessed, there was concern that users might think that an

update report gave the latest information when, in fact, it was displaying the information delivered

some time ago. To solve this problem, a statement of the time at which the page was assembled

was added to the update page. Html mechanisms for solving this kind of problem are beginning to

emerge. Client pull and server push are two mechanisms that allow the refreshing of Web pages

without any action being taken by the user. These are possibilities for future real-time information

delivery systems — especially those which are not interactive.

   TTIA was designed not to deliver useless or misleading information. For example, if a user

asks about the progress of his or her bus-of-interest which is scheduled to arrive at their stop at

5:03 p.m., the information delivery component of TTIA needs to check the current time. If it's

currently noon, it would be misleading to report any information about the bus's schedule

adherence. However, there could be an exception message in the data base for the Train number

of the 5:03 bus. That bus may be serving some other route at noon as it has the same Train

number all day. So TTIA needs to deliver an update page that tells the user that no information is known at this time.

Useless or misleading information is also a danger if the user asks about a bus that has already passed their stop. An exception message in the data base may show that the bus is 5 minutes early at the time of the user's request. If they are asking about a bus that was scheduled to arrive two hours ago at their stop, it doesn't mean that the bus was 5 minutes early then. So late requests must also be caught and answered with a reasonable message.

The TTIA system is intended to give information about small schedule deviations in normal conditions. When some unusual event occurs, TTIA may not give useful information. For example, suppose snow falls. Many busses go on snow routes. Some busses could be hours behind schedule. Riders may not want to know if their bus-of-interest is late as much as they want to know when the next bus will arrive at their stop. Or they may want to know how long it is taking busses on their route to arrive downtown, for example. TTIA is not able to deliver this information. A commercial system could be written to do so, though. As TTIA was developed, it was learned that there are many extensions that would make any deliverer of real-time information more useful. It was crucial, though, to be clear about what information was available or unavailable to report.

The BDS faces problems on a snow day, too. Since most busses will be running late, the BDS will be flooded with schedule exception messages. In order to reduce the number of messages, the dispatchers can change the parameters which control the definition of lateness. So, for example, the dispatchers may decide they need exception messages only for those busses running more than 30 minutes late. This effects the information available to TTIA. If a user requests update information for their bus-of-interest and receives an update report saying that the bus has not been reported late, they may construe that to mean that the bus is on time, when in fact it is 29 minutes behind schedule.

Another problem arises from the absence of an exception message. A bus will send an exception message only if it has on board a PCMCIA card (a removable computer component)

which carries the information about the schedule. But the bus with the card aboard may break down and be replaced by another bus. Then there will be no exception messages for that bus, as it will have no information to use to determine its schedule adherence. Worse still, the original bus may continue to send exception messages saying it is 2 or 3 hours late! TTIA should not make claims in this situation.

The same argument shows that TTIA ought not to claim that a user's bus-of-interest is on time if that bus isn't running at all. Suppose a bus never leaves the garage because there isn't a driver available. TTIA will look for an exception message, find none and report that the bus "is not known to be late." The user may well interpret this to mean the bus is on time, when there is no bus coming.

These troubles could be addressed if TTIA had access to more information about each bus. There are messages other than exception messages that are communicated to the BDS from a bus. For example, there are "health messages" that a bus sends regardless of its schedule adherence status. If a bus had broken down and never departed to serve the trip which the user has chosen as their bus-of-interest, there would be no health message from that bus. The creators of the BDS anticipated these kinds of needs, so there are also broken-down bus messages, reroute messages and other kinds of information that would benefit an elaborate real-time information delivery system.

The experience developing the TTIA system suggests that either a simple system or an elaborate system can be useful. A simple system requires more disclaimers or restriction to more sophisticated users. Many of the issues and obstacles encountered have to be addressed just as thoroughly for the production of a simple system as for a system with many features.

CONCLUSIONS

Producing a commercial TTIA-like system requires the planning and management that any software project requires. In this section advice is offered about project components that are specific to real-time applications and project components that are specific to transit applications.

The feasibility of a real-time Web-based implementation has been demonstrated by the TTIA project. The scope is one of the major concerns of an implementer. Good planning discourages time-consuming additions that are outside the original design of the system. A decision should be made early about the features that a TTIA-like system will include and exclude.

The choice of an implementer should be made with attention to the familiarity of the person with the data for the transit system. Some features of the data describing a bus system are anything but obvious to the uninitiated. Thus a transit expert is an asset worth finding.

If the organization contemplating a TTIA-like system has not offered any other real-time services to the public on the Web, network connections should be a concern early in the project. Time and expertise are needed to ensure a reliable and secure connection to the transit provider's data.

This document has described many facets of TTIA's development. It was learned, for example, that static pages are not a good choice for any system that's not small. The insights provided may steer other projects and contribute to avoiding problems. As real-time systems are certainly going to be an increasing part of Web services in the future, the difficulties and expediencies of TTIA may provide guidance.

REFERENCES

(1) Dailey, D.J. *Busview: A Regional APTS Experiment*, Transportation Research C, In preparation.

(2) http://www.its.washington.edu/projects/busview.html

(3) *Advanced Public Transportation Systems. The State of the Art, Update '96*, U.S Department of Transportation, January 1996

(4) http://www.fta.dot.gov/library/tcehnology/APTS/tws/travCll1.htm

(5) http://www smartraveler.com/

(6) http://www.itsmarta.com/Moreinfo.html

(7) *Review and Assessment of En-Route Transit Information Systems*, U.S. Department of Transportation, July 1995

(8) Vorvick, J., Dueker, K., Jackson, R., and J. Qiu, *Transit Time Internet Access, Appendix B, Feasibility of Placing Bus Route Maps on the Internet,* Center for Urban Studies, Portland State University, 1997.

(9) Vorvick, J., Dueker, K., Jackson, R., and J. Qiu, *Transit Time Internet Access, Appendix A, The Interpolated Time Module of TTIA,* Center for Urban Studies, Portland State University, 1997.
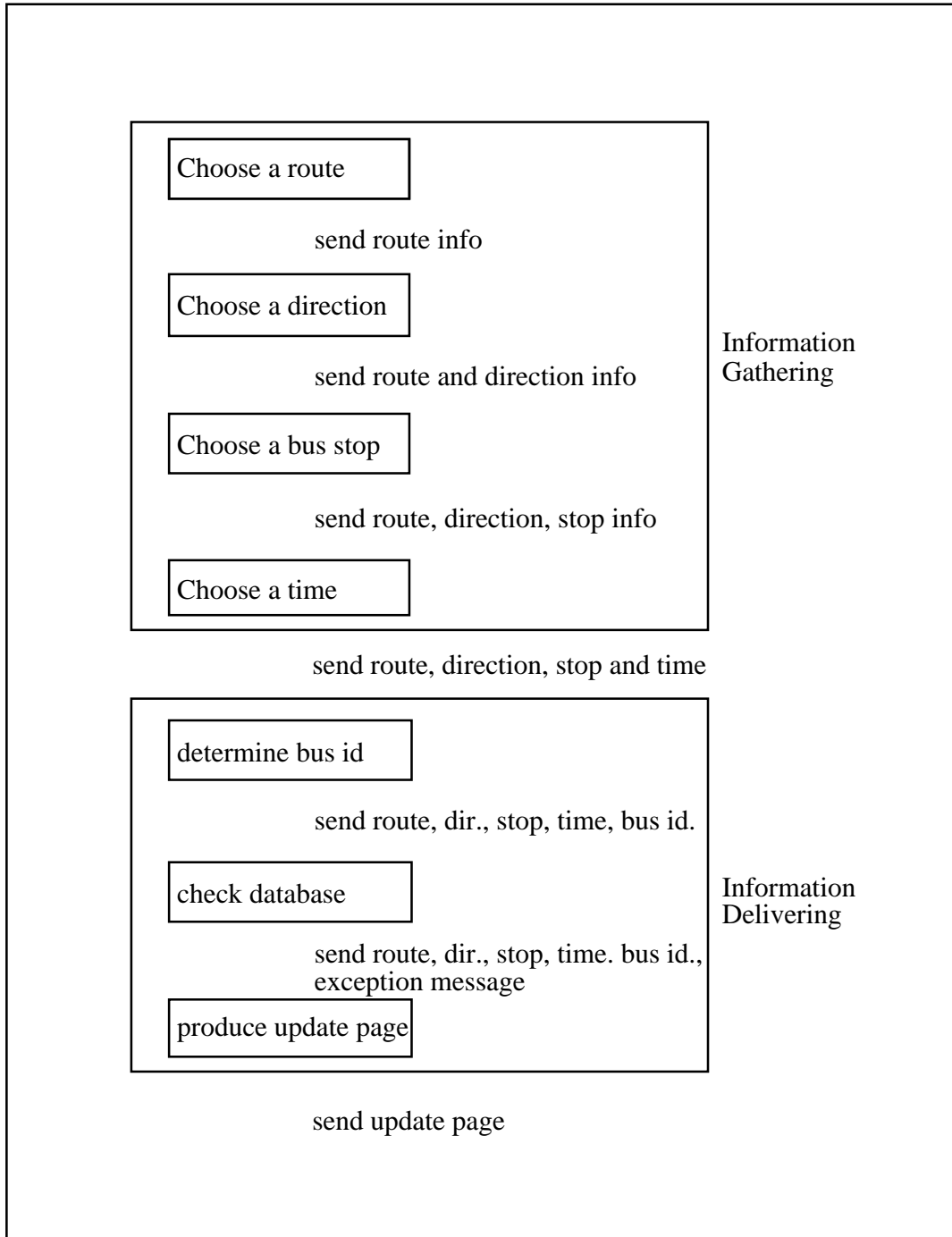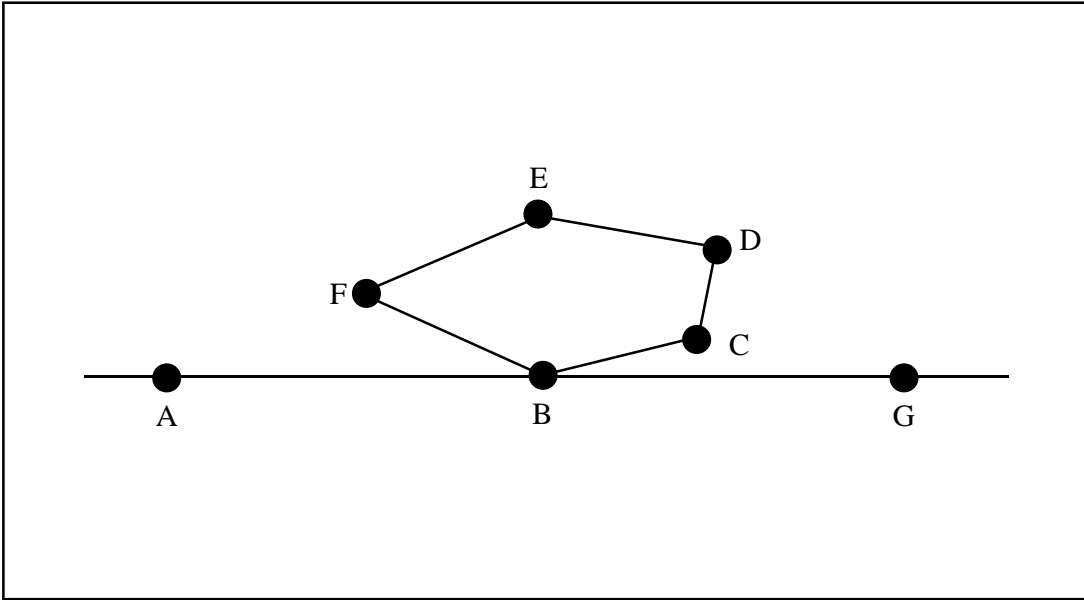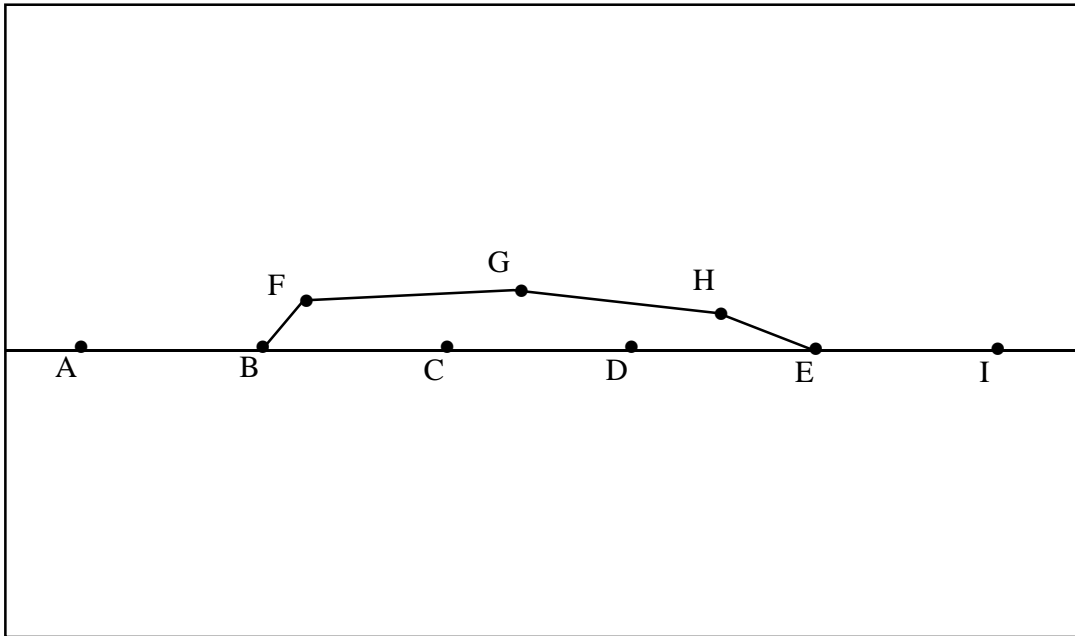
Figure 1: The Flow of Data through TTIA

Figure 2: Route with a Loop

Figure 3: Two Paths on one Route