INTEGRATING GEOGRAPHIC INFORMATION SYSTEMS INTO TRANSIT PASSENGER INFORMATION SYSTEMS

by

KAMAL T. AZAR

B.S. Civil Engineering
American University of Beirut, Lebanon (1986)

SUBMITTED TO THE DEPARTMENT OF URBAN STUDIES AND PLANNING
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER IN CITY PLANNING

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 1991

© Kamal T. Azar

The author hereby grants to MIT permission to reproduce and to distribute copies of this thesis document in whole or in part.

Signature of Author

Department of Urban Studies and Planning
May 10, 1991

Certified by

Joseph Ferreira, Jr.
Professor, Department of Urban Studies and Planning
Thesis Supervisor

Accepted by

Phillip Clay, Chairman
Department Graduate Committee
INTEGRATING GEOGRAPHIC INFORMATION SYSTEMS INTO TRANSIT PASSENGER INFORMATION SYSTEMS

by

KAMAL T. AZAR

Submitted to the Department of Urban Studies and Planning
In Partial Fulfillment of the Requirements
for the Degree of Master in City Planning

ABSTRACT

One potential application of GIS to transit management and operations is the Passenger Information System (PIS). A PIS is a computerized system used to answer passenger's inquiries about how to get from address A to address B. A PIS is generally based upon matching of addresses and shortest path concepts. This thesis:

1) explores the feasibility of providing basic PIS functions in a GIS. GIS software is usually provided with address matching and shortest path tools which might enable it to replicate many (but not all) of the main functions of a PIS.
2) examines the differences and commonalities between a GIS and a PIS. The comparison is aimed at identifying the conceptual and practical differences between a customized PIS tool and a GIS-based one.
3) defines the technological improvements, database developments, and system design strategies required for integrating PIS and GIS technologies. Both systems are based on databases which are subject to periodic updating. The recommended strategies are based on the ease of updating data and the possibility of sharing standard databases and tools with other systems in a transit agency.

To provide a concrete example, we build a prototype using Arc/Info GIS software and compare it with the PIS system used by the Massachusetts Bay Transportation Authority (MBTA).

Thesis Supervisor: Joseph Ferreira, Jr.
Title: Professor, Urban Studies and Planning.
ACKNOWLEDGMENTS

The research for this thesis was made possible in part by the support of the University Transportation Center, Region One and the Computer Resource Laboratory of the Department of Urban Studies and Planning at MIT where the work was conducted.

The Author wishes to express his gratitude and appreciation to all the people who have helped in this research, particularly to John Bottari from the MBTA, Simon Lewis from GIS/TRANS, and Professors Nigel Wilson and David Bernstein from CTS. Invaluable criticism and counsel have come from Professor Lyna Wiggins and the author is especially grateful to her.

Finally, the author would like to record his special thanks to Professor Joseph Ferreira, a devoted educator, for his supervision of this work and for his many helpful suggestions and feedbacks.
# TABLE OF CONTENTS

1 - INTRODUCTION ........................................................................................................ 8

1.1 - The Problem ........................................................................................................... 8
1.2 - What Is a PIS? ........................................................................................................... 9
1.3 - Capabilities of GIS Tools ....................................................................................... 10
1.4 - The Research Questions ......................................................................................... 12
1.5 - Significance and Importance of Research ......................................................... 13
  1.5.1 - Cost Savings / Extra Revenue ....................................................................... 13
  1.5.2 - Additional Capabilities ................................................................................ 14
  1.5.3 - Increased Flexibility / Data Sharing ............................................................ 14
1.6 - Research Methodology ......................................................................................... 15
  1.6.1 - Literature Review .......................................................................................... 15
  1.6.2 - Building the Model ....................................................................................... 15
  1.6.3 - Analysis of MBTA’s PIS ............................................................................... 16
1.7 - Summary of Thesis Organization ......................................................................... 16

2 - REVIEW OF RELATED LITERATURE ................................................................. 18

  2.1 - Automation of Telephone Information Systems .............................................. 18
    2.1.1 - Factors Used in Assessment of Telephone Information Systems ................. 20
      2.1.1.1 - Cost of the System ........................................................................... 20
      2.1.1.2 - Level of Service and Information Quality Provided to Customers ....... 21
2.1.2 - Summary of Studies Evaluating Different Telephone Information Systems 

2.1.2.1 - Early 1970’s Studies 
2.1.2.2 - Late 1970’s Studies 
2.1.2.3 - Early 1980’s Studies 
2.1.2.4 - Late 1980’s Studies 

2.2 - Route-Finding Algorithms Used by PIS’s 

3 - THE PASSENGER INFORMATION SYSTEMS USED BY THE MBTA 

3.1 - Methodology (Source of Information) 
3.2 - The Software 
3.3 - Description of Software 
3.3.1 - How Does The Software Function? 
3.4 - The Data 
3.5 - The Hardware 
3.6 - The Personnel 
3.7 - Management Plans 

4 - BUILDING A PIS PROTOTYPE USING A GIS 

4.1 - Capabilities of the PIS Prototype 
4.2 - Data Requirements for the Prototype 
4.3 - Building a Prototype PIS Using GIS Tools 
4.3.1 - First Step: Encoding the Road Network 
4.3.2 - Second Step: Encoding the Bus and Transit Routes 
4.3.3 - Third Step: Encoding Connectivity, Transfers, and Time 
4.3.4 - Fourth Step: Pre-assigning Streets to Bus Stops 
4.3.5 - Fifth Step: Address Matching 
4.3.6 - Sixth Step: Finding Shortest Path 
4.3.7 - Seventh Step: Packaging 
4.4 - Criticism of the Model 
4.4.1 - Assigning Links to Nearest Stops 
4.4.2 - Absence of Time Factor 
4.4.3 - Updating Mechanism 
4.4.4 - Segmentation of Links 
4.4.5 - Shared Arcs Between Multiple Routes
4.4.6 - Lack of Node Attribute Table ........................................... 69
4.4.7 - Limitations of ALLOCATE Module .............................. 70
4.4.8 - Limited Capabilities of Macro Language ................. 71

5 - CONCLUSIONS .......................................................................................... 72

5.1 - GIS-Built PIS vs Custom-Designed PIS ......................... 73
5.2 - Summary of Results ................................................................. 83
5.3 - Conclusion ............................................................................. 87
5.4 - Recommendations / Plausible Scenarios ..................... 89
5.5 - Future Developments .............................................................. 93

APPENDICES

APPENDIX 1 .............................................................................................. 98
APPENDIX 2 .............................................................................................. 99
APPENDIX 3 .............................................................................................. 100
APPENDIX 4 .............................................................................................. 110

BIBLIOGRAPHY .......................................................................................... 112
TABLE OF FIGURES AND TABLES

FIGURES

Figure 4.1 - Output Screen of GIS-Built Prototype ............................. 54
Figure 4.2 - Shared Arcs Between Multiple Routes ................................ 67
Figure 5.1 - Flow Chart of Macro Operations ........................................ 84
Figure 5.2 - Future PIS ........................................................................ 95

TABLES

Table 2.1 - Running Time of Shortest Path Problems .............................. 30
Table 5.1 - GIS-Built vs Custom-Designed PIS ....................................... 74
Table 5.2 - Running Time per Operation .................................................. 79
CHAPTER 1: INTRODUCTION

1.1 - The Problem

In an effort to improve the quality, productivity, and cost effectiveness of their telephone information services, many transit agencies in the U.S. have used computerized telephone information systems, called Passenger Information Systems (PIS), to provide their customers with information about schedules, routes, and fares. PIS in general are custom-designed packages which use a rich set of geographic data such as street network files and street addresses that are useful for other applications in a transit agency. However, PIS suffer generally from their lack of flexibility to share their data with other systems and to easily update their databases. Moreover, PIS do not have any graphical component: answers to queries are in textual forms only. This makes the system unable to answer requests where landmarks, and not addresses, are used as references to origins and destinations.
Geographic Information Systems (GIS) tools often include some capabilities, such as address matching and pathfinding, that can be used to build flexible PIS systems with graphical components.

This thesis examines the feasibility of using GIS technologies to build a PIS that can be used by transit agencies to assist their customers in their trip planning. First, we review the literature to understand the development and the importance of PIS and to get an idea about the technical issues related to replicating the PIS using a GIS. We then describe a typical PIS, the one used by the Massachusetts Bay Transportation Authority (MBTA), and describe various options for either adding certain mapping capabilities to a PIS, building PIS capabilities within a standard GIS environment, or constructing some hybrid combination of the two. Then we build a prototype PIS using a commercially available GIS package and compare its performance to that of custom-built PIS systems. Finally, we develop conclusions regarding the practicality of integrating PIS and GIS functions, the technological improvements that will benefit such efforts, and the system design strategies and database development efforts that can help transit agencies make effective use of GIS technologies.

1.2 - What Is a PIS?

As part of their public marketing strategy, transit agencies have developed automated telephone information systems to answer system-related questions from callers desirous of using the transit system. The customers usually call the
agency and ask about how to get from point A to point B at a particular time of
day. The customer service operator answers the phone, types the origin and
destination addresses and the desired arrival time into a computer terminal, and
the PIS package then does all the calculations that are needed to plan the trip.
The results are provided in textual form -- that is, written instructions about the
nearest bus (or train) stop, the bus to take at a specific time, any transfers that
are required, where to get off, how long the trip will take, and how much will it
cost.

PIS in general incorporates two types of databases: (1) a geographic
database such as street network files, bus routes and stops locations, and street
addresses; and (2) a transit system database such as bus schedules, fares, and
fare zones. When callers give their origin and destination addresses, these
addresses are matched against the street address files to extract the coordinates
of the two addresses and determine the nearest bus (or train) stops. The optimal
path between the two stops is then calculated based on desired time of travel and
on vehicle schedules. The query results are textually displayed and a list of the
information required for a trip is shown.

1.3 - Capabilities of GIS Tools

It is first necessary to define what is meant by the terms "GIS", "GIS tools"
and "GIS technology" that we will use in this thesis. A definition for GIS from the
National Center for Geographic Information and Analysis (NCGIA) is: "A GIS is
best defined as a system which uses a spatial database to provide answers to
queries of a geographic nature...The generic GIS can be viewed as a number of
specialized spatial routines laid over a standard relational database management
system" (Goodchild, 1985). On the other hand, by GIS tools or technology we mean
a broad range of techniques and software packages that encode spatial features,
associated attributes, and their topological relationships in more or less standard
form that can be created, stored, edited, and retrieved using generic tools for
computer graphics, computational algorithms, and interactive queries. GIS tools
have three capabilities that are useful for building PIS models:

(1) In general, GIS tools have the capabilities of reading standard street
network files such as TIGER¹, DIME², or ETAK³; or creating (digitizing) similar
files. These files contain geographic data coordinates for linear physical features,
such as street centerlines, railroads, streams, and census area boundaries, as well
as address ranges and census codes for each feature.

(2) GIS tools are able to do address-matching operations. This capability
permits the assignment of approximate geographic coordinates to addresses, the
comparison of two addresses for similarity, and the transfer of geographic
coordinates and attributes from one address to another.

¹ TIGER: Topologically Integrated Geographic Encoding and Referencing, U.S. Department of Commerce,
Bureau of the Census, Washington DC 20233.

² DIME: Dual Independent Map Encoding files, Department of Commerce, Bureau of the Census, Washington
DC 20233.

³ ETAK: street network files similar in content to DIME files, produced by ETAK, Inc., 1455 Adams Drive,
Menlo Park, California 94025.
Pathfinding is another capability of GIS technology. It is the process of calculating optimal paths between a specified origin, a destination, and any stops the route must pass through. Optimal paths are determined by finding the combination of routes that yields the shortest travel time from origin to destination.

1.4 - The Research Questions

The major questions that this research addresses are the following:

(1) Given the above capabilities of a GIS and the 'handicaps' of a PIS, is it feasible to replicate the basic functions of a PIS in a GIS? This question entails two sub-questions: (a) What kind of problems might be faced while doing this? and (b) What kind of data structure might be required for such an action?

(2) What are the differences between a custom-designed PIS and a GIS-based system? These differences can be both practical and conceptual. Two sub-questions are addressed under this question: (a) What are the practical differences between the two systems? (b) What kind of theoretical and technical aspects lie behind these differences?

(3) How practical is it to integrate PIS and GIS technologies and what are the technological improvements, system design strategies, and database developments needed for such an integration?
1.5 - Significance and Importance of Research

The importance of the research emanates from three factors: (1) cost savings / extra revenue; (2) additional capabilities; and (3) increased flexibility.

1.5.1 - Cost Savings / Extra Revenue:

The price difference between a PIS and a GIS is very remarkable. In 1988, the MBTA acquired its PIS for the contract amount of $1.5 million. Although the PIS was sold to the MBTA, its developer retained the proprietary rights for the algorithms and the data. On the other hand, the price of a GIS-based system, including the cost of the necessary hardware, training, and coding, is potentially far less than for a PIS (perhaps, less than $200,000).

Moreover, improving the accuracy of the results of the PIS may result in an increase in ridership and income to the transit agency. In a survey conducted in 1983 by the Marketing office of the Washington Metropolitan Area Transit Authority, it was reported that 82% of callers do make the transit trip for which they had requested information and that 66% of these callers would not have taken the transit trip without the information provided by the telephone information service (Ross and Soberman, 1987.) It was also estimated, based on average fares, that the net revenue per call handled by the PIS is 23.8 cents which sums up to almost $500,000 a year.
1.5.2 - Additional Capabilities:

PIS, in general, lack the capability of on-screen display of routes, while GIS tools are better geared at displaying graphical outputs. The MBTA staff believe that the algorithms used by their PIS are reliable and the information provided in response to customer inquiries is correct, however the lack of a mapping tool in the PIS prevents them from visually checking the accuracy of the results. The inability to produce maps, and the relatively complicated data updating process, in the MBTA’s PIS suggest two possibilities: the replication of the PIS with a GIS tool, or the adding of GIS capabilities to the PIS.

1.5.3 - Increased Flexibility / Data Sharing:

The difficult process of updating data in a PIS contrasts with the high flexibility of a GIS in sharing data with other systems. Many departments in a typical transit agency are potential users of GIS technology. In addition to the marketing department, which may use GIS technology to display and analyze road networks, transit routes, and bus stops in order to assist prospective riders with their trip planning, the police department could use GIS technology to map location of transit crimes or to dispatch police, and the planning department could use it to help with the mapping and route analyses involved in corridor studies. All of these applications in these different departments can use the same standard databases (TIGER, transit routes and stops) and tools (mapping, routing, and address-matching) which adds additional importance to the flexibility in data
sharing between the systems.

1.6 - Research Methodology:

To answer the three questions addressed by this thesis, the research relies on three techniques: (1) literature review; (2) building a prototype PIS in a GIS; and (3) Analysis and comparison between the MBTA's existing PIS and the prototype.

1.6.1 - Literature Review

In chapter 2 we review the literature related to the use of computerized information systems in transit agencies and summarize some reports which describe the benefits and the problems of different PIS's used in the U.S. We also examine the algorithms used in general to find the optimal path of a transit rider that are used by PIS in order to understand the technical and complex issues related to building a PIS in a GIS.

1.6.2 - Building the Model

A simplified PIS prototype was built using Arc/Info GIS software. This model is described in chapter 4. The model is not as complicated as the one used by the MBTA. It just provides the route to follow between two points. The input of the model is two addresses (origin and destination) and the output is the route to follow, including a set of route nodes (intersections) and links numbers (street
names). The factors of time, schedule variation, and bus time tables are not considered in the model for simplification purposes. The purpose of building this prototype is to check the feasibility and to surface any fundamental problems of building a PIS in a GIS.

1.6.3 - Analysis of MBTA's PIS

The PIS used by the MBTA is considered as the typical system used by transit agencies. It might be true that other agencies are using different systems which might not have the same problems as the MBTA's own, but the fact remains that many agencies are using custom-designed software which in many respects resemble the MBTA's PIS. The MBTA's PIS will be analyzed in chapter 3 to explore all the areas of the system (i.e., software, hardware, data type, updating of data, downloading and uploading of data, and personnel and management issues.) This part of the research, in conjunction with the prototype, enables us to determine the differences between a custom-designed PIS and a GIS-based system. The capability of importing and exporting data, the efficiency in providing answers to queries, the ease of updating and maintaining data, and the form of the output result are some of the factors that will be used in comparing the two systems.

1.7 - Summary of Thesis Organization

Chapter 2 reviews the literature related to automated information systems
in transit agencies and the algorithms used by PIS to find the shortest path for a transit rider between two points.

Chapter 3 is devoted to an evaluation of the PIS currently used by the MBTA. All the aspects of the system are examined to identify the problems and the ways of enhancing the system.

Chapter 4 describes the procedure of building a PIS model using a commercial GIS tool, the defects of the model, and the limitations of GIS tools to be used for building models.

In chapter 5, the thesis concludes with a discussion of the major findings of the research.
CHAPTER 2: REVIEW OF RELATED LITERATURE

In this chapter we review the literature related to the use of computer-aided customer information systems in transit agencies. In order to understand the importance of computerizing PIS’s, we look at reports which evaluate the need for, and the benefits of, automating telephone information systems in different transit agencies in the United States. We also examine the characteristics of the systems and the algorithms used by PIS’s to find the optimum path between two addresses, in order to clarify the technical issues related with replicating a PIS with the use of a GIS.

2.1 - Automation of Telephone Information Systems

Virtually all transit authorities in the US provide telephone information service to their customers in order to help them make the best use of the transit service. The extent to which this service is computerized, however, varies with the
size and complexity of the transit system. Some agencies are still counting on manual telephone systems, where the operator is responsible for determining the response to inquiries based on some types of hard-copy databases, such as street maps, route maps, schedule timetables, and other such information. The labor intensive nature of these systems and the advances in computer technology in the 1970's, however, motivated many agencies to focus their efforts on automating the information storage and retrieval within their traditional telephone system environment. This is effective since passenger inquiries are not very diverse and can be grouped into limited sets of calls manageable by computer programs.

Passenger Information Systems usually use automatic call distributor systems to answer calls or put them on hold before being routed to an agent. These systems are generally equipped with counters for counting the number of incoming calls, the answered ones, and the missed ones.

Transit agencies in general utilize dedicated mainframe computers to run these systems. In the off-hours, the computers are often used for other related activities. The applications programs for PIS systems are custom designed for every agency. Basically, these systems consist of strong database managers for matching addresses, retrieval of information, and handling data; and a set of other algorithms responsible for finding shortest paths. The geographic databases used by these systems are either standard road network files, such as TIGER, ETAK, or DIME; or digitized street networks. The PIS's used by the MBTA and South California Rapid Transit District (SCRTD) use DIME files as their database, while
the database used by the Washington Metropolitan Area Transit Authority (WMATA) was constructed by overlaying a grid of squares, 25ft on a side, on the Washington region and every bus stop, route, street, intersection, and other landmark was located by coordinates on the grid. (Diewald et. al, 1983).

Many studies were conducted to assess the usefulness of computerizing the telephone information systems in transit agencies (Wells, 1973; Shier and Gilsinn, 1978; Diewald, 1983; Phillips, 1983; Fruin, 1985; Ross and Soberman 1987). These studies started as early as the 1970's. It is interesting to note that the conclusions and recommendations of the early reports are completely different than the ones in the early 1980's. In general, the early studies did not advocate the use of computers because they deemed computers not to be cost efficient. With the advances in computer technology, the later reports supported the automation of telephone information systems.

2.1.1 - Factors Used in Assessment of Telephone Information Systems

Two factors are generally considered by these studies when assessing the importance of computerizing passenger information systems: (1) cost of the system; and (2) level of service and quality of information provided to customers.

2.1.1.1 - Cost of the System

While evaluating a system, the cost of implementing and operating the system are some of the most important factors that are considered. Many cost
elements are used for assessing and comparing systems which can be classified in the following way: initial cost, recurring costs, and dependent cost (Shier and Gilsinn, 1977). The initial costs are generally the capital costs which include computer space, computer hardware (if purchased), software development, furniture and telephones; the recurring costs are computer operation, computer hardware (if leased), and other maintenance costs (in the case of automated systems), or the operator personnel, database management, training of operators (in case of manual systems); and the dependent costs are those cost elements which depend on the number of transit information operators.

Most of these costs, moreover, can belong to more than one category. The computer hardware is one example of an item which belongs to more than one category. If not leased, an initial cost is incurred to buy the hardware; maintenance and repair of the hardware are considered as recurring costs; and the choice of hardware configuration is "dependent on the number of operators, in as much as sufficient computing power must be available to offer time-sharing service to the operator terminals without excessive wait time" (Shier and Gilsinn, 1977).

2.1.1.2 - Level of Service and Information Quality Provided to Customers

One other criterion by which alternative systems are evaluated is the level of service provided to customers. Four measures of the level of service were used by Wells (1973): average time to service a call, average time waiting to be
serviced, average time to complete a call (the sum of the service time and wait time), and the percentage of calls refused entry to the system caused by either receiving a busy signal or the queue being full.

The quality and kind of information provided to customers are two more factors by which information systems are assessed. These factors include elements like the number of alternative paths provided to customers, whether addresses only, or landmarks too, can be referenced as origins and destinations, and whether the number of transfers can be defined by a customer or not, etc.

2.1.2 - Summary of Studies Evaluating Different Telephone Information Systems

In the next sections we review some of the reports written over the last two decades about the automation of information system. The early reports were prepared before the implementation of computerized systems in any transit agency, while the later ones are after the implementation. It is worth noting that currently few agencies in the US are using automated systems, and the majority of these reports written about these systems were looking at nearly the same agencies (namely the WMATA and SCRTD).

The history of automation in public transit telephone information systems is summarized by Ross and Soberman (1987) as follows:

In the early 1970's, research carried out in several locations in the U.S. pointed to the potential applications of computer technology to data retrieval functions. Research undertaken at the National Bureau of Standards (NBS) and at the MITRE corporation investigated the costs and benefits of automated data
retrieval. Work on route-finding algorithms conducted at the System Development Corporation (SDC) in Santa Monica, California, culminated in the development of the Passenger Routing Information Systems (PARIS), a proprietary system which was successfully demonstrated by the Santa Monica Transit System in 1974. Encouraged by the earlier work and supported by federal funding, the SCRTD and WMATA pursued the development of automated transit information system technology in the form of computer-assisted telephone information systems. Although the origins of Automated Information Directory system at the WMATA can be traced to the early 1970's, system implementation did not begin officially until 1978. Software development and data entry were finished in late 1980 and system acceptance and testing was completed by February 1981. At SCRTD, a pilot implementation covering the 300-square mile San Fernando Valley region was initiated in the spring of 1977. By March 1979, the system was considered sufficiently developed. The SCRTD officially accepted the Computerized Customer Information System (CCIS) on June 4, 1979. By February 22, 1981 the CCIS was considered ready for an official evaluation of system operation and performance.

2.1.2.1 - Early 1970’s studies

As mentioned above, the early studies conducted in the 1970's did not recommend the use of computers in the telephone information systems. A study completed in 1973 (Wells, 1973) aimed at comparing three different information retrieval systems used by transit operators to search for route and schedule information, found the microfiche video system to be the most cost efficient. The three discussed systems were: (1) manual, hardcopy file system containing maps and schedules; (2) microfiche video system utilizing route and schedule information on slides; and (3) interactive computer information retrieval system. "The general result of this analysis is to reaffirm, at least for one small task, man's superiority over machine" (Wells, 1973). The analysis showed that the microfiche system was the least costly to operate and the computer system the
most costly. Although the speed of service might be increased by the use of computers, the added cost of the computer hardware offsets any savings resulting from reduction in numbers of operators. Moreover, the study showed that there is no volume of calls at which the computer would be more efficient than the other systems. The cost equation, however, showed that computers could become more cost competitive only as wages increased and computers cost decreased. The breakeven point where computers become less costly than microfiche systems was estimated to be a salary of $1900 per month. The author concluded that since "this magnitude wage rate change seems unrealistic in the near future and if such a change occurred it would most probably be accompanied by an increase in hardware costs", the microfiche system would always remain more cost efficient than any of the other systems (Wells, 1973).

In 1974, the Washington DC Metropolitan Area Transit Authority (WMATA) implemented the microfiche system to improve upon the retrieval of information operation in their system. "Initial reaction to this system was positive. But by October 1974, it was noted that the complexity of the bus and rail systems would necessitate an improved storage and retrieval system" (Diewald et al, 1983).

2.1.2.2 - Late 1970's studies

Wells' predictions about salaries and hardware were neither correct. Operators' salaries went up and hardware costs went down. It might be true that
salaries did not go as high as $1,900/month\(^1\), especially since many transit agencies use part time operators to cut their wage costs, but the hardware costs went down dramatically. These reasons, added to other factors such as complexity of networks and large sizes of databases, made some transit agencies look into the computerization of their systems. In a cost benefit analysis of automated transit information systems conducted in 1977 (Shier and Gilsinn, 1977), it was concluded that the merits to the transit company and the user from automation of the systems outweighed the potential disadvantages. However, the subjective nature of the relative benefits and disutilities associated with automation, hinders the efforts of quantification of these advantages and disbenefits. This study views the benefits and disadvantages of automation from the perspectives of the user as well as those of the transit agency.

The direct advantages of automation to users are: (1) reduction in wait time which makes information more accessible to users, and (2) consistency of computer responses which enhance the public confidence in the accuracy of the information. However, drawbacks do exist for automation including: (1) lack of flexibility in route selection in as much as provided routes might not be the best, and operators might rely on computers without looking into intelligent variations, such as walking some distance to save extra fares, (2) increased demand on the facility due to ease of access which might be translated in an increase in waiting time (as lengthy as the old waits before automation).

\(^1\) The current average rate of telephone information system operators at the Massachusetts Bay Transit Authority is around $1,700/month.
According to Shier and Gilsinn (1977), from the standpoint of the transit agency, the automation has the following advantages: (1) higher productivity of operators and more savings in operators' wages, (2) enhancement of customer relations due to the improved services, (3) improvement to transit agency image which can contribute to the attraction of more riders, (4) reduction in training cost of operators, especially since detailed knowledge of the city geography and transit routes are no longer required, (5) availability of statistics about the operation of the system, and (6) the possibility of sharing databases used in the system with other applications within the transit agency. The disbenefits of automation, however, can be summarized as follows: (1) as mentioned above, the wait time for callers may not be substantially reduced due to increase in demand and thus the image of the agency might not get any better, 2) longer response time to easy inquiries caused by the heavy reliance of operators on the computer even for the simplest queries that can be answered off their memory, 3) disruption of the service any time the computer is not operational, and 4) greater public perception of misperformance of the system as callers are given precise departure and arrival times while the system is inherently imprecise.

2.1.2.3 - Early 1980’s Studies

Many studies were conducted in the early 1980’s to evaluate the

2 As it will be seen at the end of this chapter, this is not always true. The WMATA observed that knowledge of the geography of an area is necessary to answer some queries where landmarks are used as references to origins and destinations.
performance of the computerized information systems that were implemented in the late 1970's and early 1980's by some of the big transit operators in the US, namely the WMATA, SCRTD, and MTC. A series of assessment reports were written in 1983 about the socio-economic impacts of automated transit information systems technology on the transit industry's telephone information function (Phillips 1983, 1984).

Robert O. Phillips, after examining the Computerized Customer Information System (CCIS) at the SCRTD in Los Angeles, the Automated Information Directory System (AIDS) at the WMATA in Washington DC, and the Transit Information Center System (TIC) at the MTC in Minneapolis St.Paul., concluded that (Phillips, 1984):

(1) The Automated Telephone Information Systems (ATIS) technology does work, although the development of the above three systems faced many problems, and the transition from the traditional manual systems to the automated ones took longer than expected.

(2) The implementation of ATIS, "although conducive to improved productivity, does not necessarily cause it." Theoretically, the retrieval time of information is reduced by the implementation of ATIS, but in practice, operators on the average answer up to 25% of the calls off the top of their heads, and occasionally operators take extended breaks between calls.

(3) The full support of the transit information agents and the need on the part of system designers to recognize, in addition to the technical, the practical aspects
that operators feel the system should have, are necessary for the effective deployment of any ATIS.

(4) The development and implementation process of ATIS in every agency is unique and depends on the availability of local geographic and transit databases. Moreover, due to some cost factors such as hardware procurement, system maintenance and enhancement, ATIS are only appropriate to large and complex transit agencies.

2.1.2.4 - Late 1980’s Studies

Ross and Soberman (1987), in a study aimed at evaluating the public transit automated telephone information systems, examined the experience of WMATA and SCRTD with their systems in the late 1970’s. That study almost confirmed the findings of the earlier studies, as far as telephone operators’ productivity, consistency of responses, and cost-effectiveness of systems are concerned. In the area of agents’ training, however, the anticipated benefit of training time reduction was not realistic. The WMATA, after reducing the training period to two weeks, reinstated the original six week training period as operators who were not familiar with the geographic characteristics of Washington failed to answer queries related to landmarks used as reference points during a caller’s trip. In other words, since they are not fitted with tools to show maps of routes and landmarks, the automated systems require that operators should be very familiar with the geographic details of an area in order to correctly respond to all
passenger calls.

After reviewing the importance, the advantages and the deficiencies of automating transit telephone information systems, it is important to briefly understand the functions of these systems and their capabilities in order to properly address the research questions of this thesis. Next we discuss the technical issues and algorithms that computerized transit information systems use in general. In the next chapter, we discuss in detail the capabilities of the system used by the MBTA.

2.2 - Route-Finding Algorithms Used by PIS’s

In the 1950’s and 1960’s many shortest path algorithms were developed (e.g., Bellman 1958, Dijkstra 1959, and Dial 1967) to solve routing problems. Generally, the channel that these algorithms have followed is robustness and superiority to "worst-case complexity" (Ahuja et al., 1988). These algorithms were the bases on which other algorithms were built. However, these algorithms have not been very attractive in practice as they lack efficiency.

Dijkstra’s algorithm is restricted to non-negative arc length (which is fine for transportation networks). Assume a network is made up of 'n' nodes and 'm' arcs, where $c_{ij}$ is the arc length of each arc (i,j) belonging to the network, $A(i)$ represents the set of arcs connected to node i, and C is the max arc length. Then the graph G representing the network is described as:
the way the algorithm works is briefly explained by Ahuja (1988) in the following way:

Dijkstra’s Algorithm maintains a distance label \( d(j) \) for each node \( j \) and a partition of the set of nodes into two subsets: permanently labeled nodes and temporarily labeled nodes. At each iteration, the algorithm selects a temporarily labeled node \( i \) with the smallest distance label, makes its label permanent, and scans arc in \( A(i) \) to revise the distance labels of adjacent temporarily labeled nodes. The method stops when all nodes are permanently labeled.

Dijkstra’s original implementation of the algorithm runs in \( f(n^2) \) time. Improvements to this algorithm are being performed to increase its efficiency and to handle special situations. The following table shows the different improvements to the running time of this algorithm over time. (All logarithms in this table are base 2 except for \( \log_d \), where \( d = \max(2, \lfloor m/n \rfloor) \).)

<table>
<thead>
<tr>
<th>Developer</th>
<th>Running Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Williams, 1964</td>
<td>( f(m \log n) )</td>
</tr>
<tr>
<td>Johnson, 1977</td>
<td>( f(m \log_d n) )</td>
</tr>
<tr>
<td>Johnson, 1977</td>
<td>( f(m \log \log C + n \log C \log \log C) )</td>
</tr>
<tr>
<td>Boas, 1977</td>
<td>( f(C + m \log \log C) )</td>
</tr>
<tr>
<td>Johnson, 1982</td>
<td>( f(m \log \log C) )</td>
</tr>
<tr>
<td>Fredman, 1984</td>
<td>( f(m + n \log n) )</td>
</tr>
<tr>
<td>Gabow, 1985</td>
<td>( f(m \log_d C) )</td>
</tr>
</tbody>
</table>

Source: Ahuja et al., April 1988
It is worth mentioning that the practicality, in terms of running time, of shortest path algorithms depends, in addition to the data structure, on the equipment they run on. The advancement in computer technology has enhanced the running time of these algorithms and enabled programmers to develop more sophisticated algorithms which are suitable for more complex types of applications such as finding shortest path on networks where internodal time requirements are time variable, or networks with time constraints on movement or movement prohibition (Easa, 1984; Halpern and Priess, 1973).

The shortest path algorithms used by the PIS application ideally are different than the normal standard Dijkstra or Bellman algorithms. The transit route network used by these applications have some special characteristics which make the standard algorithms inappropriate for finding the minimum expected travel time on these networks. Travel time on transit networks is random (probabilistic) and is constrained by the schedule. Buses incur random delays depending on traffic congestion, and on the demand en route. Moreover, these buses are dispatched according to schedules and the travel time (which includes waiting time) of a passenger between one origin and one destination depends on the time of his arrival at the bus stop relative to the bus schedule. In other words, travel time on transit networks is both random and time-dependent which makes standard algorithms inapplicable for determining such travel times.

Dijkstra algorithm can be used for finding the shortest path on networks with random but non-time-dependent travel times while other algorithms are
proposed a method called 'time-adaptive route choice' for finding the optimal path
for networks with travel times that are both random and time-dependent.
Basically, this method is based on the idea that "the best route from any given
node to the final destination depends on the arrival time at that node. And
because the arrival time is not known before departing the origin, a better route
can be selected by deferring the final choice until later nodes are reached." This
method is probably the most realistic in simulating the actual traveller behavior
but suffers from a long computational time as compared to the standard shortest
path algorithms. As a result, it is very unlikely for this algorithm to be useful in
PIS's as the computational and response times are some of the most important
and troublesome characteristics of such systems.

The most likely algorithms used by PIS software developers are the ones
which are to some extent realistic in replicating travellers choice and at the same
time computationally efficient. Such methods, instead of incorporating detailed
bus schedules, would estimate waiting times from bus headways (Spiess, 1986).
Another probable method might be the one proposed by Gilsinn et al (1975) which
do incorporate schedules but fail to account for the variability of travel time.

In chapter 3 we describe the characteristics and capabilities of the PIS used
by the MBTA. In chapter 4 we discuss the construction of a prototype PIS model
inside a GIS in order to compare the two systems and understand all the
complexities associated with such an action.
CHAPTER 3: THE PASSENGER INFORMATION SYSTEMS USED BY THE MBTA

In this chapter we describe in detail the Passenger Information System (PIS) used by the Massachusetts Bay Transport Authority (MBTA). In our description we discuss the functionality and the defects of the system in order to better focus the framework, and emphasize the importance, of incorporating GIS capabilities into the existing system. A brief statistical description of the MBTA’s service, at the beginning, is helpful to set the size of the MBTA with respect to other transit agencies in the US.

The MBTA services 130 towns and cities in and around the Boston Area. The total size of the service area is 2,350 square miles populated with 4.05 million persons. The average weekday total system ridership for unlinked trips is 945,000 riders. The MBTA’s basic system consists of rapid transit lines (red, orange, and blue lines), light rail vehicle lines (green lines) as well as a network of 160 bus
routes. The total number of vehicles operating on these lines, including buses, locomotives, and trolleys, is 2,230.¹

The PIS provides passengers with information about routes and modes of transport from their origins to their destinations. Inquiries are telephone-based. Passengers call the Route Information Center at (617) 722-3200 and give their origin and destination points. MBTA information operators tell them about optimum bus and train routes, and can also supply directions, fares, trip times and other helpful information.

This chapter investigates the different aspects of operation, maintenance and management of the system. Specifically, it explores 5 areas of the system:

(1) Software
(2) Data
(3) Hardware
(4) Personnel
(5) Management

3.1 - Methodology (Source of Information)

In conducting the investigation about PIS, four techniques were used to gather information:

(1) Interview: Mr. John Bottari, the marketing team leader at the MBTA, was the

¹ Source: The Massachusetts Bay Transportation Authority, Fiscal Year 1992 Budget Report, MBTA statistical Profile.
main source of information. He was interviewed on several occasions and the information he provided constitutes the bulk of this chapter.

(2) Review of the PIS manual: A quick review was done of the type, quality and quantity of documentation.

(3) Observation of PIS programs in operation: Mr. Bottari has run many demonstrations of the programs used by the PIS modules, and explained the function of each program and the way operators handle calls.

(4) Phone calls: The MBTA Route Information Center was called several times, in order to empirically investigate certain aspects of the operation of the system and the way operators answer calls.

3.2 - The Software

The software was originally developed by "Transmax", a company based in California. In October 1986, "Transmax" was bought by "Megadyne", another company also based in Santa Monica, California.

Most of the modules of the software were developed by "Transmax", but "Megadyne" combined all the modules under one package called PARIS (PAssenger Routing Information System).

The first version of this software was developed for SCRTD (Southern California Rapid Transit District), and the second for the Regional Transit District in Denver. The third and latest version is the one used by the MBTA. The difference between the earlier and the latest versions is that the former was a
unimodal package (buses only), while the latter is multimodal (bus, rapid transit, commuter rail, subway and ferry).

In April 1986, a $1.5 million contract was signed between the MBTA and Megadyne for the development of the latest version of PARIS, the installation of the software, the upgrading of the MBTA’s 4381 mainframe computer, the supply of 20 terminals, and the coding of Boston routes and lines into the software. The PIS has been in operation since the spring of 1988.

3.3 - Description of Software

PARIS is written in PL1 language. It is composed of 4 main modules which consist of almost 300 programs in total. The modules are:

1. ATDB (Address Translation Data Base)
2. RMS (Routes Maintenance System)
3. SPARTAN (Schedule Planning And Real Time Assessment of Network)
4. PARIS (PAssenger Routing Information System)

The first three modules are used for storing and manipulating data for the system, while the last consists of a set of procedures for operating the system.

The first module is used for storing and editing street addresses, locations, intersections and other geographic data. Its main data base is the DIME files provided by the 1980 Census. The RMS module is used for storing bus routes, bus stops, train lines and stations, transit lines, ferry lines and ferry docks. It is also
used for editing and updating these data. The SPARTAN module stores and edits bus and train schedules, and traveling times between stations and bus stops at different times of day. The PARIS module comprises a set of algorithms for computing traveling time and building routes between different locations based on the data stored in the other modules. Each of the above modules can be operated and used for updating data separately.

The documentation for the PIS comprises of six volumes. The first volume is the executive manual, the second is the overview for management volume, the third and fourth are the operator manuals, the fifth is the data maintenance manual and the sixth is the computer support manual. The documentation is described as "fairly good" by Mr. J. Bottari, the marketing team leader at MBTA. "Usually the manual has been useful and helpful in solving problems and explaining functions of programs", says Mr Bottari.

3.3.1 - How Does The Software Function?

The terminals used by the operators can only run some of the executable files. In other words, PIS operators have no access to data-modifying programs. Data-editing programs are only accessible by the operators' supervisor and the updating and management staff.

The program that is used most frequently by the operators is called the QRS (Query Response System) inquiry program. This program provides the routes' trip time and cost of trip between two locations. About 60% of the 2,500 daily calls
that the MBTA receive are processed through this program.

When starting, the QRS program asks for input of origin, destination, requested time of either arrival or departure, day, fare budget, any desirable mode of transport and any other preference.

The origin and destination inputs are the only mandatory ones. The rest are optional. If no departure or arrival time is entered, the departure time is defaulted to 3 min after the telephone call is received. The day also defaults to the current day if no input to the contrary is made, and the budget becomes unlimited if no entry is made for fare.

For origin and destination data, three types of input are acceptable: the origin (and destination) building number and street name, a street intersection or a landmark. If the input address refers to more than one possible location, the program displays all the possible locations that the given address might match, and asks for specifications about the required one. Landmarks include such well-known places as hospitals, universities, hotels, train stations, etc.

After the input of data, the QRS program re-displays in text the origin and destination with the complete address to make sure that they are correct. It then builds and displays the list of routes which meet the requirements. It gives the station locations, the departure and arrival times, and the travel time between stations. It also indicates the modes of transportation, the headway (destination) sign for each vehicle, the total trip time and the fare. If more than one route is possible, the program displays the trip times, the number of vehicle transfers and
the fares for all of the alternatives.

Walking time between an origin or a destination and the location where the first vehicle is to be taken is not considered. For example, if a passenger needs to go from 120 Boylston Street to the intersection of Mass Avenue and Magazine Street and wants to arrive at 5:30 p.m., the program does not consider or calculate the walking time between the Central Station and the Mass. Ave./ Magazine Street intersection which might be more than 2 minutes.

The PARIS software does not give any graphical display of routes. The routes built between two locations are displayed in the form of list of addresses and stations. This list can not be printed out, but rather a screen dump is done if necessary.

Unfortunately, the algorithms used by PARIS can not be disclosed because of a licensing agreement between the MBTA and Megadyne. However, from the results of inquiries and from the configuration file one can get an idea about the assumptions and the operations of the algorithms. One of the basic assumptions of the program is constant speed for all vehicle types. The speed of rapid transit, light rail, buses, commuter trains and ferries is assumed to be 15 mph. It can, however, be edited in the configuration file. (A list of the configuration file parameters is given in Appendix 1).

The four major components of the software are the DIME file, the bus/train routes, the bus/train schedules and the algorithms. The program does not take traffic congestion and delays into consideration. It operates in a very simple
manner. After being given the origin and destination the program tries to locate the nearest bus stop or train station to each of the origin and destination points using the DIME file, and the stops and stations file. It then builds the shortest path between these stations, based on travel time, using the routes file and the schedule file. Finally, it computes departure, arrival and travel times, based on the schedule file.

QRS has another important function. Sometimes passengers want to know the route that the bus takes between two stations. QRS can provide information about all streets taken. It can give a list of all the streets taken in addition to the turning movement at every intersection (see Appendix 2).

No major bugs have been detected yet in the software, but illogical results occur sometimes. These illogical results are not due to errors in the algorithms, but rather to factors in the configuration file. For example, in the configuration file a variable for the total trip time is given. It was first set to a large number. As a result, when a passenger would want to know when to show up at some location first thing in the morning, the program would make him travel to the nearest station the previous night, spend the night at the station, and then take the first morning bus or train at that location. This problem was fixed simply by reducing the "longest trip length" parameter so that it did not exceed 2 hours (see Appendix 1).

The accuracy of results is another issue of concern to the MBTA. There is no way to check how accurate and reliable the results are, except by getting a map
and tracing the path built by the program between an origin and a destination and judging the results using common sense.

Another defect of the model is its inability to build paths based on some constraints such as avoidance of a certain area or a certain road. Where a line is shut down for maintenance, a street detour is temporarily in force, or in emergencies, there is no way to make the program build a path which avoids taking that line, except by editing the route data files.

QRS is probably the most useful and important program in the whole package. However, some other programs are also useful to the PIS operators. PARIS has another program which gives the nearest location to a caller where he can buy a transit pass and related information such as pass price, dates of sale of passes, etc. However, at present, relatively few passengers ask for such information.

3.4 - The Data

As mentioned before in this report, PARIS requires three types of data. The ATDB module is based on the DIME file. The DIME file used by the MBTA is the 1980 release which was enhanced by a private firm, Geographic Data Technology (GDT)\(^2\), in New Hampshire.

The major problem facing the MBTA is the exactness of the DIME file used. It is believed that the DIME files are 90 % correct. Out of 90,300 records, 7,800

records are suspected to be inaccurate and 390 streets are unnamed. This results in a serious problem. Since 10% of DIME file data is "suspicious", it means that roughly one out of every ten calling passengers is given "suspicious" advice (assuming that the other databases besides the DIME file are 100% accurate).

For almost a year, a team of MBTA staff has been involved in the correction of the 7,800 inaccurate records. This task has been very tedious, as site visits were necessary to resolve ambiguous data matters on many occasions. Many of the records still need to be checked.

The format of the database used by the ATDB module cannot be given out. Apparently the licensing agreement between the MBTA and Megadyne stipulated that no disclosure of the software algorithms or data format is allowed.

The RMS (Route Maintenance System) module is based on route data files. These files are prepared by the marketing department of the MBTA. They consist of bus, rapid transit and commuter rail routes and stops. The bus routes are detailed to the level of enumerating all the streets included in a route and the turning movement of buses at every street intersection.

The accuracy of these files is, "for practical purposes, very good", reports Mr. Bottari. The percentage of error, less than 5%, is tolerable. Furthermore, the marketing department periodically updates these files and errors are reduced with every update. Although the data is being prepared by the MBTA, the format of these files is also "classified".

The SPARTAN (Schedule Planning And Real Time Assessment Network)
module is based on schedule files. Schedule files are prepared by the Schedule and Planning Department of the MBTA. These files are built for all the transportation modes used by the system.

Bus schedules are designed to accommodate the different levels of congestion for different times of day. Rapid transit, commuter rail and ferry schedules are also prepared for the peak and non-peak hours demand. These files give the departure and arrival time at every bus stop or station of all vehicles, as well as the travel time between stations at different times of the day. The format of these files is also not disclosable.

The size of these data files is large. There are 60 files in all. The number of bus routes is around 350 with about 7,500 bus stops. The amount of data makes it hard for the updating team to make the data completely error-free.

Two departments (2 persons) of the MBTA are constantly working on the update of the data. In the future, the MBTA hopes to update the bus, rapid transit route, and schedule files 4 times a year, and the commuter rail files 3 times a year. The Marketing department is planning to code more landmarks into the system in order to make the operators’ jobs easier.

As previously mentioned, the data needed for input can be a landmark, a building number and street name, or a street intersection. This type of input can serve all types of trips, (i.e, home to work, shopping, and other trips). Moreover, this type of data does not require the operators to do any part of the process in their heads, such as trying to find the nearest bus stop or train station to the
caller’s origin or destination.

3.5 - The Hardware

The hardware that PARIS is running on now is an AMDAHL mainframe. It is equivalent to an IBM 3090 mainframe. The MBTA acquired it in April 1990. It has replaced the old one initially used to run PARIS.

The PIS is not the only program running on the mainframe. It just occupies a small portion of the capacity of the machine (roughly 10%). Unfortunately, because the system has only been recently set up, few people are familiar with its maintenance and operation costs.

The PIS has 18 to 20 color terminals dedicated to it. Eight of these terminals are used by the operators, whereas the others are used by the operation and administrative staff.

3.6 - The Personnel

The PIS involves at least 25 persons in its operation, maintenance and management. Twelve operators, both part-timers and full-timers, receive passenger calls. One supervisor is in charge of the 12 operators. Two teams are responsible for preparing the route data files and the schedule data file, and two persons are in charge of updating the data.

The system operates on two shifts. During weekdays, operators start receiving calls as early as 6:30 a.m and close at 11:00 p.m. During the weekends,
they start at 8:00 a.m and close at 6:00 p.m. The system operates 7 days a week throughout the year.

On the average, between 2,000 and 3,000 calls are received per day. As a test, we called the 722-3200 several (up to 10) times. Generally, all our calls were made during the afternoon on week days. Most of the times, the line was busy and an answering machine informed us that our call was put in a queue waiting for the first available operator to help us. The average waiting time before being helped by the operator was on the average less than 30 seconds.

The QRS program is quite fast in running the query algorithm which searches for the routes, modes of transport and travel time between an origin and a destination. It takes between 6 and 14 seconds for the program to complete its search. However, a caller would normally spend at least a minute with the operator from the time he gives his request till he hangs up.

Operators are usually trained for 2 to 3 days before starting their job. They are taught how to run the QRS program and how to handle callers. No great qualifications are required from the operators. The only requirements are an ability to spell words correctly and common sense. Spelling is needed to type the origin and destination addresses correctly, and common sense is obviously needed to check the results. Slight differences in spelling can be accommodated since the PIS has some type of SOUNDEX algorithm to match addresses phonetically. Familiarity with the Boston area is helpful but is not an absolute requirement. Operators are not required to do any operation in their heads, although sometimes
they do it for very obvious searches. All they have to do is enter the addresses correctly and provide the answers to callers. As a matter of fact, the system is equipped with a SOUNDEX aid which, when unable to recognize an address, lists to the operator all the possible addresses available in the database that might match with the entered one. The operator can then confirm the spelling with the caller and pick the required one.

3.7 - Management Plans

The management of the MBTA plans to improve the PIS. Enhancements to the system, such as an increase in the number of transfers between vehicles from 3 to 4, has been carried out in the past. The main concerns of management are the accuracy of the results and the task of cleaning up errors in the data.

To check the accuracy of results, the implementation of a graphical display system is being investigated. MBTA has already bought the MapInfo software for this purpose. This software is a "low-end" mapper which runs on micro-computers. It has the capability of shading areas or coloring arcs on a map based on the values of a single variable at a time. This package can be thought of as a display system to depict the results of analysis conducted using another system. Future plans are to use MapInfo to graphically display the results of queries given by the QRS program. As of now, no such attempt has been made.

Another issue management is trying to tackle is the updating of data. Management is seeking an alternative to the manual correction of DIME files.
Plans are being made to investigate the use of the 1990 TIGER files instead of the DIME files, or the use of the TIGER files to correct and fill in the missing data in the DIME files. The reluctance in pushing forward the use of TIGER files in both cases is generated by the uncertainty about the accuracy of the 1990 TIGER files themselves.

Another future plan that the MBTA is studying is the implementation of a "voice response" system. This system will provide calling passengers with recorded responses instead of displaying answers to operators who have to read the results to the passengers. The implementation of this system is believed to improve the quality of the system in general and to decrease the answering time of operators. The number of calls handled by each operator per unit time would increase as a result.

After reviewing the way the PIS operates and its current problems, we discuss in the next chapter the feasibility of building a similar system, using a GIS, which has the basic capabilities of the PIS, and perhaps avoids the shortfalls discussed above. In chapter 5, an analysis of the built model and GIS capabilities is made, and a set of conclusions is drawn.
CHAPTER 4: BUILDING A PIS PROTOTYPE USING A GIS

In order to check the feasibility of providing basic PIS functions in a GIS, we built a prototype using Arc/Info GIS software. In this chapter we compare it with the PIS system used by the MBTA and discuss the steps of building the prototype, its functionalities, and its major defects.

The three main defects of existing PIS systems are: (1) the inability of the transit operator to generate a map showing the proposed route, (2) the difficulty in updating and maintaining data, and (3) the comparatively high cost of these tools. The first problem is potentially solved by using GIS technology to draw a map (either on the computer screen or on a printout) showing the origin and

---

1 Some of the findings in this chapter were published in a conference paper; Kamal T Azar and Joseph Ferreira, Jr. "GIS for Transit Passenger Information Systems", Geographic Information Systems for Transportation Symposium, Orlando, Florida, March 1991.

2 The total development and implementation cost of the PIS's used by the Washington Metropolitan Area Transit Authority (WMATA) and the Massachusetts Bay Transportation Authority (MBTA) are $1,635,000 and $1,500,000, respectively.
destination, the selected transit routes, transfers, and bus stops, and the surrounding streets and landmarks. The second problem (maintenance) could be helped by the same mapping tools if they allowed street segments, routes, and bus stops to be edited graphically (by pointing to the appropriate spatial feature on the screen). The third problem (cost) might be helped by building a PIS using standardized encoding of street networks and transit routes and generic mapping tools (instead of a customized, turnkey system).

4.1 - Capabilities of the PIS Prototype

To be functional and maintainable, a PIS with a mapping component must address the following capabilities:

1. **Encoding the road network** so it can be drawn to scale, used to locate street addresses, and efficiently maintained;
2. **Encoding the transit routes** so they can be drawn to scale, linked to bus stops and schedules, and efficiently maintained;
3. **Encoding connectivity, transfers and time restrictions** for the transit routes so that trips that combine routes can be constructed;
4. **Pre-assigning streets to bus stops** so it takes less time to find the bus stop that is closest to the 'origin' and 'destination' addresses;
5. **Matching addresses** for the 'origin' and 'destination' to the appropriate street segment (and, hence, to the nearest bus stop);
6. **Finding the shortest path** along the transit routes from the 'origin' stop.

49
to the 'destination' stop;

(7) Packaging the steps into a friendly application with appropriate
graphics, functionality, interaction, and speed.

These mapping and spatial analysis capabilities can be incorporated either
by adding GIS functionality to a PIS or by adding PIS capabilities to a standard
GIS package. Since it uses 'standard' GIS tools, one might expect the latter
strategy to be a less expensive way of combining graphic and PIS tools. In any
event, the cost of constructing and maintaining the required databases could be
reduced (or at least spread across more transit and highway applications) if they
were encoded in a standard format. In order to test these assumptions, we
undertook to build a prototype PIS system using, as much as possible, standard
datasets that are accessible to any transit agency, and standard network
algorithms and GIS capabilities.

4.2 - Data Requirement for the PIS Prototype

In general, the following data are required:

(1) Road network -- a digital representation of the road network in a form
that encodes sufficient positional accuracy, network connectivity, and
address information to support meaningful visual display, address
matching and shortest path computations. (The TIGER and DIME
formats are standardized and well-suited for this purpose.)

(2) Routes -- a digital representation of the bus and transit routes which is
consistent with the road network files (in terms of positional accuracy and network connectivity), but which must have additional information and attributes such as bus-only or rail-only segments (that aren’t included in the road network), connectivity restrictions (such as routes that cross without making transfers possible), and the location of stops (which need not occur only at road intersections.)

(3) Schedules -- the scheduled times, for each route, of arrival/departure of vehicles at each stop. In general, these schedules differ by day and by time of day.

4.3 - Building a Prototype PIS Using GIS Tools

While building a PIS using GIS tools is conceptually straightforward, few GIS packages come with enough tools to enable the PIS application to be built with only the "off the shelf" packages. Many software packages can display road networks (or routes) graphically but most cannot handle all the spatial analysis features listed above. Computer-aided drafting (CAD) packages, for example, could display the graphics, but generally do not represent the road segments and associated attributes in a form that is well-suited to handle address matching and routing models. Most packages that advertise GIS features are able to read (and display) road networks encoded in a TIGER format and can use these data to provide address matching capabilities. However, most such software does not
include the required path-finding capabilities, many packages lack the spatial editing and display that would be useful, and almost none of the packages provide sufficient 'macro languages' and other model building tools for the packaging of a PIS application to be a practical application development task.

Atlas*GIS and MapInfo, for instance, have the capabilities of reading TIGER files, matching addresses, and can display the maps and routes effectively. Transcad and Intergraph offer, in addition to these capabilities, some network algorithms such as finding shortest paths; but none of these packages presently have both the tools and the internal programming language needed to build the PIS application using only the 'off-the-shelf' software.

For our prototype PIS system, we used the PC and workstation versions of Arc/Info. Arc/Info is the only software package that we are aware of with enough off-the-shelf tools to permit the construction of a workable PIS. As we shall see, the tools do not enable the prototype to be packaged efficiently -- but at least it is workable.

The basic design of the prototype is to pre-process the road network and route information so that PIS capabilities (1), (2), (3), and (4) (listed above in

---

1Atlas*GIS, Strategic Mapping, Inc., 4030 Moorpark Avenue, Suite 250, San Jose, California 95117.

2MapInfo, MapInfo Corporation, Troy, New York.

3TransCAD, Caliper Corporation, 1172 Beacon Street, Newton, Massachusetts 02161.

4Intergraph, Intergraph Corporation, Huntsville, Alabama.

5ARC/INFO, Environmental Systems Research Institute, 380 New York Street, Redlands, California 92373.
section 4.1) are represented at the outset as maps ('coverages') and associated data attribute tables in Arc/Info. A prototypical PIS inquiry is then handled by supplying the Arc/Info application with 'origin' and 'destination' addresses and then waiting for the application to do the necessary computations and then draw (on-screen) a map that shows two addresses, the connecting streets, and the recommended stops, routes, and path to get from the origin to the destination. One such map is shown in Figure 4.1 for some 200+ street segments in the vicinity of Dupont Circle in Washington, D.C..

Each step of the entire process is described below -- first in general terms, and then in Arc/Info specific terms. Each step is described in two sets of paragraphs: one set of generic descriptions for readers unfamiliar with GIS technology, and one set (in italics) of Arc/Info specific descriptions for readers acquainted with GIS technology in general and with Arc/Info software specifically. (The Arc/Info commands used in each of these steps are included in Appendix 4.) It should be noted that the street segments are real, but the bus routes as well as the stops used in this figure are fabricated for the sake of building the prototype and hence are not real lines or stops. The prototype construction proceeded in the seven steps, described below.

4.3.1 - First Step: Encoding the Road Network

In this step, TIGER files are used to represent the road network geometry and topology in order to create the network map. Each row in the (first) TIGER
FIGURE 4.1 - Output Screen of GIS-Built Prototype
file contains location and attribute information for a single street segment (i.e.,
the longitude and latitude of the intersection at each end of the street segment,
and the street name and address ranges that identify addresses along each side
of that street segment). This scheme for representing the road network is useful
for matching origin and destination addresses and geographically locating
(approximately) these addresses on the map.

TIGER files come in a standard ASCII format. Arc/Info has the capability
of reading in these files, converting them into Arc/Info coverages and projecting
those coverages from latitude/longitude into state plane coordinates. These steps
produce an Arc/Info 'coverage' containing a network of arcs together with an
associated data table containing the TIGER attributes. The command TIGERARC
is used to convert the TIGER file into an Arc/Info coverage, ACREATE is used to
create an ADD (address) file, and ABUILD is used to link the ADD file to the
coverage. These three operations are needed to make the TIGER files useful for
address matching within the Arc/Info environment.

4.3.2 - Second Step: Encoding the Bus and Transit Routes

Bus and rail routes were also constructed using spatial coordinate data
from the TIGER files. They are extracted into a "routes file" from the road
network file created in step 1. The routes file is a subset of the larger road
network and hence has fewer links and nodes. Nevertheless, the extracted links
have the same geometry, links and nodes numbering as the network file links in order to ensure a desired consistency between the two map coverages and to facilitate the transfer of attributes from one to the other.

Arc/Info's ROUTE module has the capability of finding the shortest path along a network that passes through any specified set of intersections. This capability was used to generate the routes as a subset of links from the road network file. Each bus route was created by specifying all the nodes (i.e., intersections in the road network) that the route followed and then running the ROUTE command to select the links that connected all these nodes. Each route was saved to a separate coverage using the SAVEROUTE command. The motivation to build the routes from the road network was to enable the address range information for a street segment to be related easily to the route segment that ran along that same street.

Since the number of links and nodes which make the route coverage are fewer than the original number in the TIGER file, cleaning a route coverage would cause the renumbering of nodes and links and the loss of linkage between the TIGER coverage and the route coverage. To get around this problem, the links of the TIGER coverage which do not belong to a particular route are deleted in ARCEDIT while keeping all the nodes. Then the newly created route coverage can be converted into a usable ROUTE coverage using the BUILD command. (CLEAN
4.3.3 - Third Step: Encoding Connectivity, Transfers and Time

When a GIS is used to represent streets as (center) lines, the intersection of two links generally implies connectivity. (This is because GIS tools tend to focus on 2-dimensional 'planar' geometry that doesn't explicitly account for the geometry of 'overpasses'.) However, in the case of transit, connectivity does not necessarily exist at any intersection of two routes. For instance, a rail line might be crossed by a bus route where no transfer is possible at that intersection. To accommodate this possibility (and other turn restrictions that might complicate the path that one can take through an intersection), one must have some additional capacity to store information (e.g., in tables) that are linked to specific nodes (intersections) in the map and data files that contain the routes and the road network.

In Arc/Info, information about turn restrictions are stored in separate turntable files. A turntable contains the node number, from arcs, to arcs, azimuth, and angle between arcs for a particular intersection in the coverage. A time impedance item is added to this coverage and calculated based on the angles and azimuths. The time impedances are read by ROUTE and ALLOCATE modules to

---

* Version 6 of Arc/Info is currently being developed. This version has greatly expanded 'route' handling capabilities that, among other things, will probably enable the creation of node attribute tables which can be used to avoid this complex set of steps needed to retain the association between road and route links.
simulate the waiting time at street intersections. The usefulness of a turntable is that it allows one to set explicitly the degree of connectivity between arcs. If two arcs cross each other and are not connected (e.g., an overpass, or a bus line crossing rail line), the turntable can be used to set a negative impedance for the right and left turns and hence forbid connectivity between the two arcs. The drawback of the turntable is that they are keyed to node numbers which can change if the road network changes.

4.3.4 - Fourth Step: Pre-assigning Streets to Bus Stops

For simplicity, we assume that people always begin their transit trip at the nearest bus stop to their origin and then get off at the nearest stop to their destination. (Other more complex options are discussed later.) In order to speed up the PIS process, every street segment is pre-assigned a 'closest' stop. This allocation operation is run beforehand and its results are added to the attributes of the road network file. As a result, every address can be associated with the nearest stop to it much more rapidly than if we had to recompute which stop is closest every time we ran the PIS application.

The ALLOCATE module is used in Arc/Info to determine the nearest stop to every address. It has the capability of finding the closest stop to each street by determining all of the optimum paths from each stop and assigning all the streets along those paths to each stop. The ADDCENTER command is first used for
designating the stops, followed by RUN and GROW commands to execute the allocation and display the results. WRITEALLOCATE saves the results of the allocation to the TIGER file as an attribute in order to be used for other applications. It creates a new attribute field, where nearest stop numbers are saved as attributes to the links.

4.3.5 - Fifth Step: Address Matching

The road network file is used to locate the origin and destination addresses. GIS tools, in general, are provided with the capability of matching an address to the TIGER street segment containing the address. In order to integrate this capability into the PIS application, however, the results of the address match must be supplied in a form that can be used in the next step (i.e., finding the id of the nearest stop to that street segment). Just because a package can highlight on-screen the street segment containing the matched address, it may not be able to pass the id of that link along to another command needed to accomplish the next step.

In Arc/Info, the origin and destination addresses can be matched against the road network file to create a third file which consists of the attributes of the links (including the nearest stop) to which the matching addresses belong. Two files are required: an info file where addresses to be matched are stored, and an address coverage. The address coverage is the one created from the TIGER file, while the
The info file is defined in TABLES. After reading records from the info file, the program searches for matching records in the coverage file, and creates a new point feature coverage for the matching records. The point coverage for the matching records contains the internal number of the arcs to which the addresses matched. This number is used to link the matching coverage to the TIGER coverage.

4.3.6 - Sixth Step: Finding Shortest Path

After identifying the nearest stops to the origin and destination addresses, the shortest path between these two stops is calculated. GIS tools have the capability of finding the optimal path between two nodes which belong to a network. The optimal path is determined based on the lowest waiting and travel time associated with all possible paths between the two stops. The routes file created in step 2, the connectivity file created in step 3, and the nearest stops found in step 5 are used to calculate the optimal path. In addition, GIS tools are used to display the location of the stops and the path connecting them.

The ROUTE module in Arc/Info has the capability of finding the shortest path between any two points. The route coverage is used to calculate the shortest path between these two stops. ADDSTOP, ADDROUTE, and PATH commands are used for this purpose.
4.3.7 - Seventh Step: Packaging

A scripting or 'macro' capability is needed in order to link the above six steps and to use the output of one operation as the input for another. The basic strategy we adopted is to run steps one to four in advance, and then to package steps five and six into a macro. The macro asks for input of origin and destination addresses, matches them to the street network file (created in step 1), and identifies the nearest stops assigned to the streets where the addresses match. It then uses the routes file (created in step 2) to calculate the shortest path based on the delays at intersections and the connectivity file (created in step 3) between the two stops. It finally draws the routes network, marks the locations of the stops, and highlights the streets which belong to the shortest path.

The macro, in Arc/Info, is used to ADD to the info file the origin and destination addresses, match the info file to the address coverage, identify the internal number of the links where addresses matched, find the nearest stops attached to these links from the TIGER coverage, and use these nearest stops to determine the shortest path between them. Finally, a series of display commands are used to draw the required features, labels and routes on-screen (as shown in Figure 4.1).
4.4 - Criticism of the Model

The model is built on many assumptions, and the updating of routes in it is quite a sophisticated process. The defects of the model can be attributed to either the limitations of GIS tools in general and/or to Arc/Info’s specific capabilities. The following is a list of assumptions that were convenient but questionable and a discussion of several other limitations of the model:

4.4.1 - Assigning Links to Nearest Stops

It is assumed that every address has only one stop close to it and this stop can be used for travel in any direction or any distance. In fact, more than one stop might be close to an address and different stops might be good for different travel purposes or different directions. For example, it might be that one address is located between two bus stops or is close to a train station and a bus stop. The train station is usually the one useful for long trips while the bus stop is a better choice for short ones. Moreover, the closest stop to an address might not always be the best for all kinds and directions of travel. In this model, however, every address is assigned only one stop regardless of the direction of travel, and the direction and the type of travel are not taken into consideration while allocating stops to links.

One way of reducing the crudity of this assumption, would be the assignment of two types of stops to every link: "local stops" and "express stops". Local stops can be the nearest bus stops and light rail stops, while the express
stops can be the nearest heavy rail and train stations. The allocate command can be run twice to allocate to every link a local stop and an express one. The shortest path between each of the two origin near stops and each of the two destination near stops can be calculated, and the shortest one of the four possible combinations would be considered the best.

In order to accommodate for the time-of-day variations in transit routes, the model can be taken to a higher level of sophistication by extending this allocation method to local and express bus stops by time-of-day. The allocation command can be run for every time of day variation of the network and assign a nearest stop (local and express) to each link for each of these variations. The selection of nearest stop, can be then dependent on the time travel. This issue is further discussed in the next section.

4.4.2 - Absence of Time Factor

The model does not take into consideration any time-of-day factor. It also assumes that in-vehicle travelling time or travelling distance are the only factors which determine the shortest path. Waiting time is not given any consideration. In reality, bus schedules vary across the time of day and the best morning path between two addresses might be different than the evening one for the same addresses. Moreover, the standard shortest path algorithms used by GIS software, such as Dijkstra's algorithm, are not the appropriate ones in networks with travel times that are both random and time-dependent. More adequate algorithms, such
as the "time-adaptive route choice" (Hall, 1986), should be used instead.

Also, the assumption that distance of a stop from an address is the only criterion for selecting a stop is not correct. The level of service of different lines is an important factor that needs to be considered too. That is, the headway between buses at a stop which is within 5 minutes walking distance from an address might be much larger than the one at another stop which might be seven minutes walking distance from that address, for example.

Incorporating time factors into the model would render the task of coding it into a very complicated job. A simplified way of incorporating time would be to include waiting time and delays in the turntable. The expected waiting time at every stop (which can be approximated to half the headway) can be added to the delay time for every stop in the turntable. To accommodate for the variation of the schedules, three delay times can be created for every stop: morning peak, evening peak, and off-peak delays.

4.4.3 - Updating Mechanism

The defects in the updating mechanism can be attributed mainly to deficiencies in current state-of-the-art GIS tools. Any time a change to a route is done, almost the whole process of building the model has to be repeated. This is caused by the way the prototype is built. In this prototype the routes file and the streets network file have to be always consistent as far as node numbering is concerned. For any change to the routes network therefore, the whole process of
extracting routes from the streets network file has to be repeated. On the other hand, as the sharing of street network files becomes more and more common, it will be increasingly desirable to have an updating mechanism which can properly integrate different sources of data. For example, it would be useful for the GIS to have the capability of extracting all the links and nodes from a regionally maintained street network file (with say, TIGER format) just by snapping an outside coverage or "cookie-cutting it" (i.e., spatially extracting) into a new coverage where the node-ids and link-ids remain congruent to the main file.

For the prototype, the way data is entered into a turntable requires some manual work. If no connectivity exists at certain crossing lines, the setting of negative impedances must be manually done. A better way of reducing the level of manual work while coding connectivity would be to add to the turntable coverage two sets of attributes related to the type of "from-arc" and "to-arc" lines. A negative impedance can be set when these types are different, (i.e., when the from-arc is a rail line and the to-arc is a bus line or vice versa). However, manual work would be still required to set transfer points, but to a lesser degree, if such global coding of connectivity is done. It is worth mentioning that Arc/Info allows such change to be automated, but the prototype did not include such code.

4.4.4 - Segmentation of Links

In a GIS, bus stops should be located at nodes (simply for convenience of network analysis). Nodes in TIGER files, however, represent street intersections
only. Bus stops are not always located at streets intersections but sometimes along a street between two intersections. If bus stops are to be inserted by splitting the arcs, the GIS would assign all the attributes of those links to both parts of the split arcs. For example, the same address range would be assigned to both arcs while in reality it should be rearranged proportionally to the length of split arcs. As a result, when matching addresses, more than one location would match a certain address which would disturb the running of the program. Furthermore, the splitting of arcs would create new link internal id-numbers and new node numbering which would cause the loss of link between the TIGER coverage and the old routes coverage, unless the route coverage is re-built on top of the updated TIGER file. However, the problem of re-arranging address ranges would remain unsolved, unless manual changes of addresses are to be done. It might be argued that the re-arranging of addresses can be easily automated by assuming that addresses are evenly spread along a street. However the node numbering issue would remain complicated.

This problem is an example of a more general segmentation problem when data from different sources are merged. Shared data affect significantly the way the database is designed. Different applications requirements have to be met by the same dataset which contributes to the complexity of the segmentation problem. For example, the representation of a route can differ from one application to another. If a branch of a route is made up of a set of links connected in series to each other where no stop exists along these links, for instance, the
whole branch needs to be aggregated into one link for the efficient computation of shortest path, while no such representation is tolerable for address matching applications. (The data sharing issue and its effects on data design is further discussed in the next section).

4.4.5 - Shared Arcs Between Multiple Routes

A GIS does not handle every route as an entity by itself, but rather as a set of links and nodes. In other words, routing programs in a GIS are built on link and nodes concepts. A shortest path program, for example, would check every link connected to a node while calculating a route. Some of the links connected to a node might not be a part of a route. The turntable is used to set a negative impedance for these links to inform the program to exclude them while calculating shortest paths. The turntables, however, do not function properly in the case of shared links between multiple routes. This idea is better clarified in the following example:

![Diagram of shared arcs between multiple routes]

FIGURE 4.2 - Shared Arcs Between Multiple Routes
Links a, b, & d belong to route 1, while links c, b, & e belong to route 2. Link b is common for both routes. The turntable can be used to set negative impedances at node n, for flows coming from link a to link c and vice versa in order to restrain the flow from going in the wrong direction. The flow on the b link can be either the one coming from route 1 or route 2. The turntable at node n, however, cannot be used to inform the program that only the flow coming from link a can turn to link d and only the flow coming from c can turn to link e. At n, the flow should be allowed from b to both d and e. As a result, flow coming from link a can go to link e and the one from link c can go to link d. As it might be the case that no stop exists at node n, the GIS would hence generate wrong routes when a link belongs to multiple routes.

This problem arises since the topology of routes is not just the geometry of streets. One way of avoiding the problem would be to 'double-encode' every link used by more than one route. Other techniques, such as using the data side of the GIS, would also be efficient in avoiding such problem. Note that the complexity of this matter emanates from the difficulty in sharing these data if such encoding is intended to serve the purpose of multiple applications. The double-encoding strategy for instance, is not acceptable for graphical applications such as the production of schedule maps by the scheduling and service planning departments in a transit agency. Hence the data design task is not a simple job but must provide a carefully constructed encoding scheme that can be augmented to support the different needs of several applications. While complex, appropriate database
design can be simpler than coordinating the maintenance of multiple versions of road and route networks.

The next 3 sections of this chapter discuss the limitations of the model caused by Arc/Info's specific characteristics. The issues discussed in these sections therefore might not apply to other packages or can not be generalized to GIS technology in general.

4.4.6 - Lack of Node Attribute Table

The current version of PC Arc/Info lacks the capacity of storing data related to nodes. Nodes in a GIS can only be identified by referring to the links connected to them. If nodes are deleted or added to a coverage, after CLEANing that coverage these node numbers will be reshuffled and matching the old node numbers to the new one is almost impossible. It can be argued that turntables can be somehow considered as node attribute tables, as they contain "From-arc-id" and "To-arc-id" items which can be used to relate between coverages. While this is true, it is problematic for the following reasons:

(1) The relational logic needed to trace 'from' and 'to' arcs through nodes is best handled in SQL tools that are generally not yet available from within GIS packages.

(2) One-to-many problems are very likely to occur if links are used to identify

---

*The next release (version 6.0) of Arc/Info promises to provide node attribute tables.*

69
nodes. A simple example would be two links forming a ring and connected by two	nodes (or pseudo-nodes).

(3) If a node is relocated to another position while still connected to the same
links, it might be considered to be the same node whereas, geographically, it is
another one.

4.4.7 - Limitations of ALLOCATE Module

The ALLOCATE module is used generally to assign resources (links) to
centers (stops). Centers in Arc/Info should be always located at nodes. In TIGER
files, nodes represent street intersections only. Therefore, in the ALLOCATE
module, bus stops can be created at street intersections only. In a city, however,
it might be true that most of the bus stops are located at street intersections, but
there are many stops located along some of the links. The problem of creating
nodes at these locations is discussed above (Section 4.4.4). Relocating stops at
intersections might be an inaccurate (but to some extent acceptable) approximation. The ALLOCATE module has the capability of locating addresses
but can only start the allocation algorithm from a node. Starting at an intersection
results in a simpler algorithm whereas starting at the middle of a link requires
a more sophisticated algorithm. This issue raises the more general question about
how rich a set of algorithms to expect to have bundled in with a standard GIS
package.
4.4.8 - Limited Capabilities of Macro Language

The macro languages used by Arc/Info are somewhat limited (although the workstation AML is less limited than the PC SML). For instance, they both lack a general capability of setting the output of one operation as the input for another. The AML does it for few commands though awkwardly. Hence, the results of an operation have to be written into a file which can be read by another operation as the source of input. These operations of opening files, writing, reading, and closing them result in a low efficiency in terms of running time. For example, ARCPLOT has an AML command that allows attributes of a selected feature to be stored directly into a macro program variable without having to write an intermediate file. However, going into ARCPLOT to take advantage of this capability is dependent on the type of platform and on the software design of Arc/Info. The next chapter contains a detailed discussion of these topics.

The time required by the model to determine one route is almost 130 seconds on the PC. We built another version of the model on the workstation to enhance the running time of the prototype. The running time on the DECstation 3100 is 40 seconds. As discussed in the next chapter, the improvement in the workstation running time of the PC and workstation prototypes may mainly be attributed to the difference in platform, different macro languages, and the software engineering of Arc/Info and the operating system (DOS vs. Unix). A more detailed comparison between the customized model and the GIS-based system is included in the conclusion chapter.
CHAPTER 5: CONCLUSIONS

In this chapter, we compare the model we built based on a standard GIS to the customized one used by the MBTA in order to address the research questions of whether adding certain mapping capabilities to PIS or building PIS functionalities within a standard GIS environment is the right track for enhancing current PIS. We first discuss the meaning of building a PIS inside a standard GIS, then the limitations of current "out of the box" GIS software, and finally we give our conclusions and recommendations.

In determining what does it mean, or what does it take to build a PIS in a GIS, a comparison between the model we built and the custom designed one used by the MBTA, is made. In this comparison the performance of each model is considered and the running time difference between the customized and the GIS-based one is examined in more detail. A general conclusion is made about the trade-offs of adding a GIS to the current PIS's, versus building a PIS on top of a
As mentioned before, the question of whether Arc/Info is an appropriate tool or not, can not be easily answered, especially since some GIS software that overcomes some of Arc/Info's limitations, also currently lacks some of its other capabilities. Some of the latter capabilities proved quite useful for the PIS example. These capabilities include the ability of encoding transit routes and stops, matching addresses, finding shortest path, packaging, etc (for more details refer to chapter 4). As we mentioned in chapter 2, on the other hand, at least 3 large transit operators in the US are using a similar PIS system or an earlier version of the same system used by the MBTA, namely the Southern California Rapid Transit District (SCRTD), the Washington DC Metropolitan Area Transit Authority (WMATA), and the Regional Transit District in Denver. Generalization from the MBTA's system to PIS's is therefore permissible.

5.1 - GIS-Built PIS vs Custom-Designed PIS

In the comparison of the GIS-built model to the customized one the following issues are considered: network size, running time, form of output, data ownership, data maintenance, platforms, software design, and estimated cost (see Table 5.1). The initial PC prototype was too slow to be practical for real-time phone usage. We converted the model to run on a workstation (DECstation 3100) in order to check whether or not its efficiency improved. The results were significantly improved, as is illustrated at the end of the following discussion.
(1) Network Size: The network used by the MBTA's PIS is based on DIME files and consists of more than 90,000 links, around 350 bus routes (including variations), and 7500 stops. The GIS-built model is based on TIGER files and consists of 200 links, 4 routes, and 12 stops.

(2) Form of Output: The output of queries in the customized system is displayed in the form of text and consists mainly of stops and bus route numbers, and times of travel. In contrast, the query results of the GIS-built models are both graphical and textual. A map showing the origin and destination addresses, the nearest stops, and the routes is drawn.

<table>
<thead>
<tr>
<th>ISSUES</th>
<th>GIS-BUILT MODEL</th>
<th>CUSTOMIZED PIS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network Size</td>
<td>200 Links</td>
<td>90,000 Links</td>
</tr>
<tr>
<td></td>
<td>4 Routes</td>
<td>350 Routes</td>
</tr>
<tr>
<td></td>
<td>12 Stops</td>
<td>7500 Stops</td>
</tr>
<tr>
<td>Running Time</td>
<td>130 sec (PC)</td>
<td>10 sec (Amdahl)</td>
</tr>
<tr>
<td></td>
<td>40 sec (DECstation)</td>
<td>10 sec (Amdahl)</td>
</tr>
<tr>
<td>Form of Output</td>
<td>Graphic and Text</td>
<td>Text only</td>
</tr>
<tr>
<td>Data Ownership</td>
<td>Public</td>
<td>Proprietary</td>
</tr>
<tr>
<td>Data Maintenance</td>
<td>Flexible</td>
<td>Difficult</td>
</tr>
<tr>
<td>Platform</td>
<td>IBM PS2 / DEC 3100</td>
<td>Amdahl Mainframe</td>
</tr>
<tr>
<td>Software Design</td>
<td>Standard Tool</td>
<td>Customized</td>
</tr>
<tr>
<td>Modularity</td>
<td>Front-End to Standard Tools</td>
<td>Black Box</td>
</tr>
<tr>
<td>Estimated Cost</td>
<td>?</td>
<td>$1.5 Million</td>
</tr>
</tbody>
</table>
(3) **Data Ownership:** Although the MBTA paid around $1.5 million for their system (including the hardware), the developer of their PIS owns the proprietary rights for the code and the data structure used by the system as well. Such a problem can be avoided if a GIS-built model can input and output data in a standard non-proprietary formats in which much data are already available (e.g., TIGER files).

(4) **Data Maintenance:** The lack of graphical tools and the overlaying capabilities in the customized PIS are the major handicaps to the data maintenance process. The PIS used by the MBTA, for instance, is fitted with some tools for the updating of the databases. However, almost 10% of the DIME file records used by the system are believed to be inaccurate. As a result, for the last two years, the MBTA’s marketing department has worked on updating these records. In the DIME files, for example, many of the streets cut through buildings or are connected to other streets where in reality no connection exists. Field checks are needed to identify these records. The overlaying capability of the GIS, however, would save the trouble of site inspections in many situations. As more municipal agencies adopt GIS, accurate street network maps that will help edit transit routes will be accessible to a PIS tool that handles standard GIS data formats. For example, the map of Boston, at the block level (digitized by the city from aerial photography and transformed into a format readable by standard GIS tools), could be overlayed on top of the existing DIME file to facilitate the identification and
updating of wrong records. Moreover, the GIS-based system can more readily accommodate standard road network data that could be shared across applications in the transit agency or across agencies.

(5) **Platforms:** The MBTA's PIS runs on a mainframe (Amdahl), while the GIS-built models run on an IBM PS2 Model 80 PC, and on a DECstation 3100. The MBTA's PIS is a multi-user system which can run many queries simultaneously by accessing the same databases and the same programs from different terminals. The PC version of the GIS-built prototype is a single user system. The programs and the databases are accessible by just one user at a time while the workstation version of the prototype is capable of supporting multi-user functions although some parts of the macro would have to be changed to avoid conflict over filenames. By Multi-user PIS, we mean a system where the database, programs, and graphics components are accessible and executable by different users from different terminals at the same time. The GIS-built model (workstation version) is characterized by having all its components accessible by different users simultaneously but having some of its programs executable one at a time. The graphics component, for instance, can be accessed by different users at the same time and maps of the same region can be drawn simultaneously. However, many of the intermediate steps in the macro require the use of intermediate files whose names must be altered to accommodate different simultaneous users. This was not done for the prototype. If multiple users ran the current prototype simultaneously,
they would be writing and deleting the same files and would get unpredictable results.

(6) **Software Design:** The Customized PIS used by the MBTA is very efficient at performing routing queries, but lacks the flexibility of integrating, or being integrated into, other systems. The proprietary rights of the builder, however, refrained us from investigating the software design details. The GIS model, in comparison, is built as a "front-end" to standard GIS tools. The address-matching, finding of shortest path, allocation of resources, and graphical display capabilities are standard/general purpose GIS tools that can be used to build a PIS model. In Arc/Info these tools are packaged in different modules and a macro language is used to create the link between the different modules. This modularity added to the weakness of the macro language contributes to the low efficiency, in terms of running time, of models built on top of these modules.

(7) **Cost:** Cost comparisons are difficult since the prototype was not designed for multi-users and high transaction operations and since very little data are available from the MBTA about the cost details of their system. It would have been interesting to breakdown the cost into initial, recurring, and dependent as suggested by Shier and Gilsinn (1977), but for the same reasons such a comparison is not possible. Also, the cost and benefit associated with each option can be direct or indirect. The GIS-built models have the advantage of cheaper
initial cost, but as illustrated above, the GIS tools are not geared at specifically performing the PIS functions and hence are not as efficient as the customized systems, which have additional indirect cost. On the other hand, the cost of adding a GIS to a PIS in order to be used as the tool for the graphical display of query results might be negligible compared to the initial cost of a customized PIS. However, the benefits gained from such an operation may not be worth the effort if the "black boxish" feature of the system remains unchanged and the ability to update and share data remain very complex.

(8) Running Time: In the custom-designed PIS, the average running time of a query is around 10 seconds, compared to 130 seconds on the PC GIS-built model, and 40 seconds on the workstation model. To understand the difference in running time we had to examine where the time is spent: is it on the effective computational time of the programs or is it on the overhead caused by the modularity and the structure design of Arc/Info? As the PC version of the model is far from being efficient (and hard to improve due to the limitation of the PC macro language and the speed of PC's in general), we examined the workstation version in more detail. In the macro used to run the program, we inserted time measurement commands for every major operation in order to keep a record of where the time is being spent\(^1\). Essentially, three major operations are executed every time the macro runs. Each of these operations is run in a different module:

\(^1\) This time stamping step added a few seconds of overhead to the entire operation.
the address matching is run at the ARC level, the identification of the nearest stop in the ARCPLOT module, while the calculation of the path is in the ROUTE module (see appendix 4 for the workstation version of the macro used to run the model). The time toll for each operation is shown in table 5.2.

<table>
<thead>
<tr>
<th>OPERATION</th>
<th>RUNNING TIME [seconds]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Observation #</td>
<td>1 2 3 4 5 6 7 8 9 10 AV</td>
</tr>
<tr>
<td>Loading INFO Module</td>
<td>3 4 3 5 3 4 5 3 5 3 3.6</td>
</tr>
<tr>
<td>Creating INFO File</td>
<td>2 1 1 1 2 1 1 1 1 1 1.2</td>
</tr>
<tr>
<td>Cleaning Old Matching File</td>
<td>5 6 5 5 4 4 4 4 4 4 4.6</td>
</tr>
<tr>
<td>* Address-Matching</td>
<td>6 6 7 6 7 5 6 6 7 5 6 6.1</td>
</tr>
<tr>
<td>Loading ARCPLOT Module</td>
<td>10 6 11 9 10 10 11 4 4 4 8.5</td>
</tr>
<tr>
<td>* Finding Nearest Stop</td>
<td>3 2 3 3 3 2 3 2 2 2 2.6</td>
</tr>
<tr>
<td>Loading ROUTE Module</td>
<td>10 4 10 10 9 9 9 5 5 5 8.0</td>
</tr>
<tr>
<td>* Calculating Optimal Path</td>
<td>7 8 8 7 8 7 9 7 9 7 9 7 7.6</td>
</tr>
<tr>
<td>TOTAL RUNNING TIME</td>
<td>46 37 48 44 48 44 43 44 37 31 42</td>
</tr>
</tbody>
</table>

* Basic operations of the prototype

The total running time varied from 31 to 48 seconds. This is due to the fact that other users’ processes were running simultaneously and affected whether needed code was already in RAM (or swapped out) and whether RAM could accommodate all the required processes of Arc/Info. It can be seen that the most variation in operation time is in loading modules (e.g., between 4 and 11 seconds for loading ARCPLOT) which is dependent on availability of free swap space and
memory. The macro involved many overhead operations such as loading modules into the memory, killing old files, and moving from one module to another. These operations consume more than 60% of the total running time of the program.

The running times of the basic operations required for finding the shortest path (excluding overhead operations), however, are worth examining in more detail. The address-matching operation took on the average 6 seconds, the finding of the nearest stop 3 seconds, and the path-finding 8 seconds. In total, the time required for running the whole program, excluding overhead, is 18 seconds.

The 18 seconds figure, however, is not very useful for comparison between the MBTA’s PIS and the GIS-built prototype because of scale issues. As mentioned above, the sizes of the two networks are very different. The number of arcs in the MBTA’s PIS is almost 450 times larger than the prototype, while the number of routes is almost 40 times larger. The number of arcs affects the address-matching operation while the number of routes affects the path-finding algorithm. In order to make the address-matching operation efficient on large scale networks, indexing by street name, street type, address prefix, and suffix of all records can be done. The running time of this operation would, as a result, remain within an acceptable operational time.

The time for finding the nearest stop is almost independent of the network size, since the nearest stop is pre-allocated to every arc in the network. Hence, when an address is matched, the nearest stop is automatically identified. The 3 seconds spent on this operation is for selecting the arcs that matched from the
database and reading the nearest stop from them. As the database size increases, the selection process is slightly slowed down but still remains very fast.

Out of the 8 seconds spent on finding the shortest path, almost 4 seconds are spent on reading the network and on plotting it. These times can be considered as graphic overheads and a better software design can eliminate these delays (see next section for further details about improved software design). The actual computation time of optimal path is therefore about 4 seconds. As seen in chapter 2, the running time of the shortest path algorithm is dependent on the number of nodes and links in the network. Unfortunately, no information about the type of algorithm used in either system is available to estimate the running time of the prototype if applied to large networks. However, it is likely that the running time would increase at a high rate (probably \( m \log_2 n \), where \( m \) is number of arcs and \( n \) is number of nodes) as the number of nodes and arcs increases which makes standard shortest path algorithms unpractical and inefficient for PIS use. The prototype includes 69 arcs for the routes while the MBTA's PIS has approximately 24,000 arcs. Assuming the number of nodes is equal to the number of arcs, then the prototype could take 830 times longer (24,000 \( \log_2 24,000 / 69 \log_2 69 \)) to find the shortest path on the full MBTA's system. Assuming that half the 4 seconds time for the prototype is not a once-only setup step, the full scale prototype could take (2 x 830) 1,660 seconds to find the shortest path-- i.e., half an hour. Since routes have relatively few intersections and shared links, relative to the total number of arcs, this worst-case bound is
likely to be a substantial overestimate. Nevertheless, it is clear that the special structure of routes would have to be exploited for the algorithm to perform acceptably on MBTA-scale problems.

As discussed in chapter 2, standard shortest path algorithms are not entirely appropriate to be used on networks with travel times that are schedule-dependent and random. This inadequacy, coupled with the inefficiency described above, advocate one of two options: (1) the customization of these algorithms to make them useful for PIS type applications, or (2) the use of the database side of the GIS to provide for the functionalities that are lacking on the geographic side of current GIS.

The customization of standard algorithms is one alternative for computing efficient shortest path. Efficient codes which use more adequate and sophisticated algorithms than the standard ones can be written outside the GIS. These codes can access the databases used by the GIS to calculate shortest paths and output the path results in a form readable by the GIS. This approach, it might be argued, runs against the purpose of using GIS in passenger information systems since it suggests the use of customized algorithms in order to solve problems initially associated with the use of customized systems. However, general purpose GIS cannot be equipped with algorithms for all types of application and the sophistication of typical GIS algorithms is far less than what some commercial routing packages use.

The second option suggested for handling the inefficiency and the
inappropriateness of standard shortest path tools used by GIS is the use of the database side of the GIS. "Relational Data Base Management Systems (RDBMS) permit the use of the GIS's geographic database with other data sources not automatically recognized by GIS systems and other application-specific customizations,... and permit the building of network analysis tools" (Grayson, 1991). Shortest path tools can be built on the database side of GIS and can be sophisticated enough to account for all the time variations in schedules. The execution times of such algorithms are quite high, but with sensible indexing, and the use of powerful workstations the performance of such algorithms might be enhanced.

5.2 - Summary of Results

The most significant factors affecting the slow performance of the GIS-built model are the modularity and the Macro languages of Arc/Info. As seen in Figure 5.1, three modules of Arc/Info need to be used in the model and the results of operations in one module need to carried over as input for other operations in other modules. The PC macro language used by Arc/Info does not allow the setting of the results of any operation as variables, but rather the results need to be written into a file in order to be read by other operations. The writing and reading of files involves the opening and the closing of these files before and after every operation which delays the running of the programs. In other words, a major contributor to the low efficiency of models built on top of Arc/Info is the fact that
FIGURE 5.1 - Flow Chart of Macro Operations

**TABLES**
- Recreate Info Address Table
  - Purge Old Data From Info File
  - Add New Addresses To Info Table

**ARC**
- Input Origin/Destination Addresses
- Rewrite ASCII Address File
  - Kill Old Coverage of Matching Addresses
  - Write New Addresses Into ASCII File
- Match Addresses
  - Create Point Coverage of Matching Addresses
  - Create ASCII File With Street Lines

**ROUTE**
- Find Nearest Bus Stop
  - Find Nearest Stop From Street File
  - Create ASCII File With Stops
- Draw Network
- Read In Stops
- Calculate & Show Path

**TIME %**
- 35%
- 25%
- 15%
- 25%
the macro language cannot call functions (except for a few commands on the workstation version), pass parameter values, and return results in the style of most high level programming languages (without having to store the results in files). In the prototype we built, the ARCPLOT module was used not for plotting or displaying the network, but for setting the output of operations in the INFO module as variables to be used in other modules without the need for writing the results into files. The reason for using ARCPLOT is simply that it is the only module where the right command for selecting fields from a table in INFO and directly setting them as variables exists. However, the number of operations that can assign their results directly to variables is very limited.

The modularity in the software structure of Arc/Info is another contributor to the delay in running time. Moving from one module to the other is the most time consuming. This can be observed in Table 5.2 and Figure 5.1. Table 5.2 shows that, in the workstation version of the model, calling up modules and loading them into the memory constitute the majority of the overhead in terms of running time of the model. Similarly, Figure 5.1 shows that most of the time on the PC version of the model is spent loading and moving between modules.

The sequential processing of operations is another one of the inefficiencies

\[\text{2 In fact there are some other indirect and complicated ways for selecting fields from a table and setting them as variables, such as going into INFO, creating a "sub-macro" inside INFO which assigns a field to a variable when run outside INFO, quitting INFO, and running it at the ARC level. Interestingly enough, the performance of this macro varied significantly between platforms. When run on a IBM RT 120 workstation, this macro was extremely slow (200 seconds) since loading INFO into the memory of the machine consumed 30 seconds on the average. The same macro however, had a good performance on DECstation 3100. The average running time of the whole macro was around 30 seconds. This illustrates the variation in terms of efficiency of programs between platforms.}\]
of current GIS software. In Arc/Info for example, if a program requires the use of more than one module (e.g. Info, Route, and Arcplot), the modules cannot be introduced in advance and already running when the queries and operations are needed. Each module must be started and stopped sequentially. No two processes can be run at the same time. "Multi threaded" programming capability will enable GIS software to be more efficient. Either the GIS software must be rewritten or the operating systems must enable some form of parallelism, or at least some efficiency in retaining programming modules in memory, for easy and fast re-use.

In summary, the current structure, modularity, user interface, and macro languages of GIS software are adversely affecting the choice of building a PIS on top of current GIS tools. As was illustrated before, these factors render the "off the shelf" GIS inefficient in handling complex operations, such as building a PIS, which require the use of more than one module and the transfer of results between programs. As GIS technology becomes more unbundled and modular, the need for improved internal programming languages will increase. Together with advances in workstation technology, these changes will make it much easier to build efficient tools for PIS type applications. Most likely, such applications will enable complex and modular routing algorithms to handle PIS routing problems while using data and displaying results through convenient links to standardized GIS packages.
5.3 - Conclusions

The first question that this thesis addresses is the feasibility of, and the problems associated with, replicating the basic functions of a PIS in a GIS. By building the prototype, the possibility of replicating PIS functionalities is proven but the many assumptions made for that purpose suggest that the current raw GIS tools are not sufficient to make the PIS viable. However, this research has illustrated the importance of, and set the guidelines for, building PIS systems using standard GIS tools in order to overcome the deficiencies of current PIS.

PIS consultants ought to be aware that GIS tools, with some improvements (discussed above), can become more viable than the existing turnkey PIS systems especially that such GIS-based systems will be less expensive and more flexible in accommodating and sharing data from different sources. GIS developers, on the other hand, ought to enhance their systems in order to make the building of PIS-like applications feasible on top of their systems. It might be argued that the market for PIS applications specifically is small enough\(^3\) not to encourage the addition of more sophisticated algorithms to standard GIS packages. However, should the system prove to be efficient and not expensive, medium and small transit operators might adopt it. Moreover, a GIS developer might use this capability as a steppingstone for entering the transit agencies' market where other GIS capabilities, such as mapping, buffering and overlaying are required. (For

\(^3\)The number of large transit operators in the US who have large and complex network of transit routes and can afford to spend hundreds of thousands per year on the use of computerized information systems is less than 50.
more details, refer to the above discussion and the description of the prototype in chapter 4).

The difference between the customized PIS and the GIS-built one and the advantages of using either one are the second subject of research of this thesis. The primary advantage of building a PIS on top of a GIS is flexibility. Obviously the ability to visualize query results and the comparative ease in editing data are the direct benefits of using GIS in general. In a transit agency, where the same data is used for different applications in different departments, the data sharing issue is of great importance. The use of GIS to build a PIS allows the importing/exporting of updated network data from/to other applications which saves much duplication of work between departments and results in the standardization of data coding across the agency. It is important to reiterate that, unlike the case with customized PIS, data ownership should not be an issue of concern if GIS is used to build a PIS. (For more details, refer to chapter 3 where the MBTA's PIS is fully described, and to the above section where a comparison between the MBTA's system and the GIS prototype is made).

The last question that this thesis investigates is the technological improvements and system design strategies required for the integration of GIS technologies into PIS. The issues of modularity, macro language, and sequential processing are the major handicaps of current GIS tools which forbid the building of efficient PIS types of applications. To make GIS tools more practical for constructing PIS applications, operating systems which support multi-threaded
programming applications, combined with better software engineering which overcomes the modularity, macro language, and sequential processing problems are required. Finally, customization of the shortest path algorithm or the use of the database side of the GIS are also necessary to make the integration of GIS in PIS more efficient and real-time useful.

5.4 - Recommendations / Plausible Scenarios

The natural questions that follow our long discussion of the use of GIS in Passenger Information Systems and the shortfalls of both current PIS and GIS tools, are: What should transit agencies now do? Which of the two systems should they adopt? Should they try to develop their own PIS using current GIS's, should they buy a custom-designed software tailored for their own needs, or should they wait until GIS technology meets all the requirements discussed above? There is no single answer to these questions, as the answers are related to many factors and characteristics of a particular agency. These factors include the budget that each transit agency has allotted for its PIS, the management attitude towards technology, and the current level of use of GIS and standard databases at the agency.

We believe that, based on these factors, transit agencies should follow one of the three following paths. These paths are ordered from least complex to most complex.
(1) "Get Ready": Agencies who have not computerized their information systems yet and have low financial capabilities need to be prepared for the implementation of a "GISed" PIS at a later stage. The pathway that these agencies should follow is to use standardized databases across the agency and start collaborating with other agencies for the sharing and updating of common databases. Standard street network files, such as TIGER, can be used to code transit routes and stops in a uniform way across the agency for the different activities of each department. The importance of standardization of databases lies in the ability to easily maintain and update these records and in the avoidance of any duplication of work (Azar and Ferreira, 1991). Digitized maps, for example, should be transformed into standard coordinate systems in order to permit the sharing of these maps with other agencies and utility companies.

Furthermore, the standardized database is useful as the foundation for building a PIS. If the agency chooses to develop its in-house PIS later on, the standardized data will enable them to build a model free of most of the previously discussed common PIS defects. If the agency, on the other hand, decides to contract out the building of its PIS, the set up for a system that can share its data with other programs or systems will be already established for the contractor. When contracting out, those agencies should avoid slipping into other agencies' mistakes. For instance, attention should be paid to the issues of ownership and proprietary rights of programs and data structures. Transit agencies should reserve their rights at least for the data and data structure used by any model
built for them by a contractor. Moreover, as emphasized above, these models should be built on standard databases or at least in standardized formats and should have the flexibility of incorporating or being hooked up to other standard models or tools.

(2) "Make Do For Now": Transit agencies who have tight budgets and have already invested large amounts in PIS's which lack the flexibility of maintenance, sharing data, and graphical output (such as the MBTA), need to find ways of improving their systems with the least expenditure. As discussed before, the graphics component is very helpful in improving these systems. It is very useful for graphically updating geographic databases. For example, by displaying a street network on a screen, updating of data can be done by pointing at an arc or a node to display the attributes attached to it and then changing them. The graphical component, however, is hard to incorporate directly into the customized systems due to the 'black-boxish' feature of these systems. In this case, the graphical component can be a part of a separate system which has some GIS capabilities (which can be a low-end GIS or a CAD package) where the data is down-loaded from the PIS into the separate system for maintenance and up-loaded back into the PIS after being updated. Unfortunately, this task is not an easy one, but at least it improves the previous process of updating data. The problem of sharing data can also be resolved if the customized PIS is based on some standardized databases (such as the DIME files in the case of the MBTA or the SCRTD). The
graphical system, however, has to be the link between the PIS and the rest of the systems. This is likely to always be a cumbersome but manageable process that is useful until the agency develops more general capabilities to support road and route network files.

If the PIS system is not based on any standard data format or the geographic data is digitized into the system, conversion programs should be written to transform these data into standardized ones in order to make them exchangeable with other data from other sources. If digitized maps are not based on real world coordinates, moreover, these maps should be registered and transformed to some standard coordinates, such as state plane coordinates or longitude/latitude coordinates, in order to make them sharable with other applications.

(3) "Go For It": This path is recommended for transit agencies with a strong capital budget and progressive management that are interested in the future development of technology in general and PIS's in particular. These agencies need to build, or contract out the building of, a PIS which utilizes standard GIS tools (address-matching, and proper transit path finding, graphical display) and standard databases (TIGER or DIME). The system should handle multi-tasking operations and have the capabilities of overcoming all the problems of modularity, script language, and multi-threaded processing. The system should take advantage of the advances in hardware and operating systems development in
order to resolve the above problems. In other words, the PIS should have the capability of importing and exporting data from and to other systems in order to facilitate the problems associated with maintenance and updating of records. For example, street network files, if standardized, can be shared between transit agencies and other utility companies.

Standard tools are also a necessity. Routing tools should be useful, in addition to finding transit routes between two addresses, for performing other routing applications such as dispatch of patrol operations which are required by transit police, for instance. The graphical component of the system should also be an updating tool and a general purpose mapping and plotting tool. In addition to displaying routes and road networks on screen, the graphical module should have the capabilities of updating databases from the screen and updates should be automatically saved into the database. The graphics module, furthermore, should have the ability to produce maps that are used by the PIS that are useful to planning and scheduling departments. All of these tools should be run in an environment where multi-tasking is possible. Moreover, the operating system where these modules run should support multi-threaded operations in order to permit the future building of efficient and sophisticated tools.

5.5 - Future Developments

Advances in computer and transit technologies will certainly have great impacts on PIS systems. Automatic Vehicle Locator (AVL) technology is being
introduced into a number of transit properties in North America to monitor location of buses. Global Positioning Systems (GPS) are also being investigated for this purpose too (Dueker, 1990). Computing capabilities of workstations technology is becoming more powerful and faster. New hardware capabilities are becoming more and more used.

PIS systems that can take advantage of all these advances might result in significant change in the nature and the quality of the service provided by PIS. AVL systems can be integrated into PIS to provide new services similar to the ones provided by teleriders⁴ and to update on-the-fly bus and train schedules according to real-time information. Such an integration will improve the quality and the accuracy of the information provided to customers.

Workstation advances and windowing capabilities will help make the delivery of such information convenient and efficient by enabling multiple types of information to be used simultaneously on screen. PIS's can become real trip-planning programs. They can be used to plan a whole trip for a caller instead of providing him or her with directions only. For instance, customers might call and list their proposed activities without necessarily knowing specifically where their needs are best fulfilled. The operator can provide the caller with the options that are available, in addition to the directions and the other information provided by traditional PIS. (Figure 5.2 shows a screen sample of such a proposed system).

---

⁴Teleriders are systems designed to provide callers with information about bus arrival times in real-time at specific stops.
FIGURE 5.2 - Future PIS Model
For example, callers can tell the operator that they are interested in first
going to meet somebody at some particular place, going together to the nearest
shopping mall (finding the closest mall would be part of the trip planning) to buy
"item X", and then returning home by supper time. The operator would have in
one window the map of the area showing the transit routes. In a second window,
a database manager program connected to a dataset of all land-uses in the area
where malls, theaters, restaurants, hotels, etc. can be queried by name or activity.
In a third window, the operator would have an open line to the supervisor in case
of any need for help. The operator needs to: (1) find the nearest mall from the
place the caller is meeting the other person, (2) check the database whether "item
X" is sold in any shop at that mall, and (3) enter the addresses of the caller, the
meeting place, the nearest mall, and the destination, in addition to the time
restrictions specified by the caller. The program could display the proposed route
and the trip description. The operator could visually check the correctness of the
calculated trip before conveying the directions to the caller. While elaborate, such
enhanced services might be sufficiently valuable to customers to justify the use
of a 900 number that will enable some cost recovery.

We believe that the future PIS's would have the characteristics of the
system described above (or at least some of them) such as the graphical module,
the multi-tasking capability, the windowing capability, and would use standard
databases and tools that can be shared with other activities. Based on the current
trend of advances in technology, we believe that such a PIS will be in use by
transit agencies before the end of this century. PIS developers will be aware of the importance of these characteristics and will try to outbid each other by taking more advantage of the technology development which will result in better Passenger Information Systems.

Finally, we believe that future research will focus on finding more "exotic" ways of providing public transit information to potential riders. Research work will aim at providing the above information through interactive cable TV or remote connection from computers to the PIS system where passengers will perform their own queries, just as the operators do, without any need for assistance.
**APPENDIX 1**

P560 426A KISP560 S560B *** MBTA *** 05/18/90 15:27:58

**QRS CONFIGURATION MAINTENANCE - SCREEN 2 **

SCREEN: 2

**CHARACTER AND DEBUGGING PARAMETERS**

USE LOAD FACTORS? (Y/N): Y

**SEARCH AND ITINERARY CALCULATION PARAMETERS**

INITIAL STOP RADIUS (PARCELS): 6  RADIUS INCREMENT (PARCELS): 4
MAX RADIUS : 48  MX RADIUS(2PASS SALES) : 16
MAX WALK DIST (XFERS) : 2
MAX WALK DIST (XFERS-CBD) : 1  MX STOP WLK(CBD-1/64 MI.UNIT): 12
MAX DRIVE DIST PARK/RIIDE : 32
DELAY FROM CURRENT TIME(MINUTES): 3
NO. PREV/NEXT STOPS IN 'STOP' : 6  NO. PASS AGENCIES RETRIEVED : 6
MIN WAIT MINUTES AT XFER POINT : 3  MIN WAIT MINUTES AT XFR (H/C): 4
ASSUMED SPEED OF TRIP IN MPH : 15  LOCAL TRIP LENGTH IN MILES : 1
PARCELS PER MILE : 16  LONGEST TRIP LENGTH (MINUTES): 120

ENTER-LOOKUP PF1-EXIT PF2-PREVIOUS PF3-NEXT
PF5-UPDATE PF7-CONFIRM PF11-CONFIG MENU CLEAR-RESET

98
APPENDIX 2

P500 426A KIS500 S500A *** MBTA *** 05/18/90 15:53:10
ROUT  *** QRS INQUIRY ***
ORIG: START / /
DEST: / /
WHEN--LV/AR: TIME: DAY: FARE:
LINES: 137S PREF: ACC:
LINE: 137 S
FROM: STA : READING
TO: STA : MALDEN CENT
1. PROCEED (WEST) ALONG LINCOLN ST
2. TURN RIGHT (WEST) ALONG WOBURN ST
3. TURN RIGHT (EAST) ALONG MAIN ST
4. TURN LEFT (EAST) ALONG WASHINGTON ST
5. TURN RIGHT (EAST) ALONG VILLAGE ST
6. TURN RIGHT (SOUTH-EAST) ALONG JOHN ST
7. CONTINUE (SOUTH-EAST) ALONG JOHN ST
8. TURN LEFT (EAST) ALONG RICHARDSON AV
9. TURN RIGHT (SOUTH) ALONG MAIN ST
10. TURN RIGHT (SOUTH) ALONG BANKS PL
11. TURN RIGHT (SOUTH) ALONG MAIN ST
12. TURN RIGHT (WEST) ALONG PLEASANT ST
13. TURN RIGHT (NORTH) ALONG S WASHINGTON ST

(LAST PAGE)
APPENDIX 3

READING IN TIGER FILES

1 : TIGERARC PRECENSUS WASHTIG TIGER.DAT

2 : PROJECT COVER WASHTIG WASHTIG
   : INPUT
   : PROJECTION GEOGRAPHIC
   : UNITS DD
   : QUADRANT NE
   : PARAMETERS
   : OUTPUT
   : PROJECTION STATE PLANE
   : UNITS FEET
   : PARAMETERS
   : END

3 : BUILD WASHTIG LINE

4 : ACREATE WASHTIG LINE # 4 WASHTIG.ACODE
   : LEFTADD1
   : LEFTADD2
   : RGTADD1
   : RGTADD2
   : FDPRE
   : FNAME
   : FTYPE
   : FDSUF
   : END

5 : ABUILD WASHTIG LINE

6 : CLEAN WASHTIG WASHTIG2

100
CREATING ROUTES FROM TIGER

1 : ROUTE
2 : READNET WASHTIG2
3 : DISP 4
4 : MAPE WASHTIG2
5 : DRAWNET 4
6 : DRAWNODE 3
7 : ADDROUTE 1 3
8 : PATH 2 1 5 7 10 15 20 24 32 37 38 49 53 55 64 79 83 99 98 38 49 53 50 55 64 79 83 99 98 97 112 111 118 123
9 : WRITEROUTE ROUTE NEXTARC
10 : ADDROUTE 2 5
11 : PATH 2 4 8 12 13 17 18 23 31 46 47 56 60 63 69 68 73 91 105 111 118 123
12 : WRITEROUTE ROUTE NEXTARC
13 : ADDROUTE 3 6
14 : PATH 93 84 81 71 65 62 57 42 43 44 45 46 47 48 49 53
15 : WRITEROUTE ROUTE NEXTARC
16 : ADDROUTE 4 10
17 : PATH 93 100 101 102 103 87 88 89 80 89 90 91 92 82 83
18 : WRITEROUTE ROUTE NEXTARC
19 : Q
20 : COPYCOV WASHTIG2 WASHTIG4
USING ARCEDIT TO CREATE A ROUTES COVERAGE

21 : ARCEDIT
22 : EDITCOV WASHTIG4
23 : EDITFEATURE ARCS
24 : SELECT FOR ROUTE = 0
25 : DELETE
26 : Y
27 : QUIT
28 : BUILD WASHTIG4 LINE
29 : BUILD WASHTIG4 POINT
CREATING A TURNTABLE FOR WASHTIG4

1 : TURNTABLE WASHTIG4

2 : ADDITEM WASHTIG4.TRN WASHTIG4.TRN TIME 4 12 F 3

3 : TABLES

4 : ARC

5 : SELECT WASHTIG4.TRN

6 : RESELECT ANGLE LT 45 AND ANGLE GT -45

7 : CALCULATE TIME = 0

8 : NSELECT

9 : RESSELECT ANGLE LE -45 AND ANGLE GE -135

10 : CALCULATE TIME = 0.15

11 : NSELECT

12 : RESELECT ANGLE GE 45 AND ANGLE LE 135

13 : CALCULATE TIME = 0.25

14 : NSELECT

15 : RESSELECT ANGLE GT 135 AND ANGLE LT -135

16 : CALCULATE TIME = 0.5

17 : NSELECT

18 : Q STOP

103
CREATING TIME IMPEDANCE FOR WASHTIG4 COVERAGE

1: ADDITEM WASHTIG4.AAT WASHTIG4.AAT SPEED 4 12 F 3
2: ADDITEM WASHTIG4.AAT WASHTIG4.AAT FT_IMPED 4 12 F 3
3: ADDITEM WASHTIG4.AAT WASHTIG4.AAT TF_IMPED 4 12 F 3
4: TABLES
5: ARC
6: SELECT WASHTIG4.AAT
7: CALCULATE SPEED = 30
8: CALCULATE FT_IMPED = LENGTH / ( (SPEED * 5280) / 60 )
9: CALCULATE TF_IMPED = TF_IMPED
10: Q STOP
ALLOCATING ARCS TO STOPS
1 : ALLOCATE
2 : READNET WASHTIG2 LENGTH LENGTH
3 : DISP 4
4 : MAPE WASHTIG4
5 : DRAWNET 3
6 : ADDCENTER 2 1320 # 0 5
7 : ADDCENTER 13 1320 # 0 5
8 : ADDCENTER 15 1320 # 0 5
9 : ADDCENTER 45 1320 # 0 5
10 : ADDCENTER 49 1320 # 0 5
11 : ADDCENTER 62 1320 # 0 5
12 : ADDCENTER 47 1320 # 0 5
13 : ADDCENTER 83 1320 # 0 5
14 : ADDCENTER 87 1320 # 0 5
15 : ADDCENTER 93 1320 # 0 5
16 : ADDCENTER 111 1320 # 0 5
17 : ADDCENTER 112 1320 # 0 5
18 : ADDCENTER 123 1320 # 0 5
19 : RUN
20 : GROW
21 : WRITEALLOCATE NEAR_STOP
22 : Q
ADDRESS MATCHING

1 : TABLES
2 : ARC
3 : DEFINE ADDRESS-FILE
   : ADDRESS
   : 45
   : 45
   : C
   : <Enter>
4 : SELECT ADDRESS-FILE
5 : ADD FROM ADDRESS.ASC
6 : Q STOP
7 : AMATCH ADDRESS-FILE ADDRESS WASHTIG2 MATCHOUT
8 : TABLES
9 : SELECT MATCHOUT.PAT
10 : LIST WASHTIG2#
11 : (take note of washtig2# for both records (181,20))
12 : SELECT WASHTIG2.AAT
13 : RESELECT FOR WASHTIG2# = 181 OR WASHTIG2# = 20
14 : LIST NEAR_STOP
15 : (take note of NEAR_STOP for both records (15, 87))
16 : Q STOP
SHORTEST PATH BETWEEN STOPS

1 : ROUTE
2 : READNET WASHTIG4 FT_IMPED TF_IMPED TIME
3 : DISP 4
4 : MAPE WASHTIG4
5 : DRAWNET 4
6 : ADDSTOP 15
7 : ADDSTOP 87
8 : ADDROUTE 15
9 : PATH 15 87
10 : LISTNODES
11 : Y
12 : QUIT
PC VERSION OF MACRO

&rem ********************
&rem ***** PIS.sml ******
&rem ********************
cls
&type " WELCOME TO MINI PASSENGER INFORMATION SYSTEM "
&type ""
&type "(Use Hyphens or Underscores Instead of Space While Typing Addresses)"
&type ""
&response 21 " ENTER YOUR ORIGIN ADDRESS :" 1447_16th_st
&response 22 " ENTER YOUR DESTINATION ADDRESS :":
1308-New-Hampshire-av
&openw address.asc I
&write %21
&write %22
&write
&closew
&type ""
&type ""
&type "RUN address.sml IN TABLES"
&type ""
&type ""
&tables # address
kill matchout all
amatch address-file address washtig2 matchout
&type ""
&type ""
&type "NOW RUN pick.sml IN TABLES"
&type ""
&type ""
&tables # pick
&type ""
&type ""
&type "FINALLY RUN draw.sml IN ROUTE"
&type ""
&type ""
route draw
arc
sel address-file
purge
y
add from address.asc
q stop

arc
sel matchout.pat
dump matchout.asc delimited washtig2#
&open matchout.asc
&read 1
&read 2
&close
sel washtig2.aat
reselect for washtig2# = %1 or washtig2# = %2
dump stops.asc delimited near_stop
q stop

readnet washtig4 ft_imped tf_imped time
disp 4
mape washtig4
drawnet 4
drawnodes
&open stops.asc
&read 11
&read 12
&close
addstop %11
addstop %12
addroute 1 5
path %11 %12
listnode
APPENDIX 4

WORKSTATION VERSION OF PIS PROTOTYPE MACRO

/* *************************************
/* *********** PIS.AML ***********
/* *************************************

&type \ 'WELCOME TO THE MINI PASSENGER INFORMATION SYSTEM'
\
&s 21 [response ' ENTER YOUR ORIGIN ADDRESS ' '1447 16th st']
&s 22 [response ' ENTER YOUR DESTINATION ADDRESS ' '1308 NEW HAMPSHIRE AV']
&s add1 := [open address.asc status -w]
&s writ1 := [write %add1% %21%]
&s writ1 := [write %add1% %22%]
&s clos := [close %add1%]
&type \ ' Run address in TABLES' \
/* *************************************
/* *** Write address into Info File ***
/* *************************************

&data arc info
arc
sel ADDRESS-FILE
purge
yes
add from /mit/maps/pis/address.asc
q stop
&end
/* *************************************
/* **** Find the Matching Addresses *****
/* *************************************

kill matchout all
addressmatch address-file address washtig2 matchout
&type \ ' Now Run pick in ARCPLOT' \

110
/* *******************************
*/
/* ***** Pick the Nearest Stops *****
/* **********************************/

arcplot
&s .a [show select matchout point 1 ITEM washtig2#
&s .b [show select matchout point 2 ITEM washtig2#
reselect washtig2 arcs washtig2# = %.a% or washtig2# = %.b%
&s .x1 [show select washtig2 arc 1 ITEM NEAR_STOP]
&s .y1 [show select washtig2 arc 2 ITEM NEAR_STOP]
q

/* *******************************
*/
/* ***** Find The Shortest Path *****
/* **********************************/

route
readnet washtig4 ft_imped tf_imped time
disp 9999
mape washtig4
drawnets
addstop %.x1%
addstop %.y1%
addroute 1 5
path %.x1% %.y1%
listnode
BIBLIOGRAPHY


Collura, J., R. Bonsignore, and P. McOwen, "Computerized Management Information


Huxhold, W. E., Modernizing Land Information Management Systems for City Planning and Management: Problems and Opportunities. (Washington DC, 1989).


