University of Southampton

Engineering an Extensible Model for a Public Transport Journey Planning System

by

Garrett Philip Fingerle

Thesis submitted for the degree of Doctor of Philosophy at the Faculty of Engineering and Applied Science

February 2001
UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING AND APPLIED SCIENCE

CIVIL AND ENVIRONMENTAL ENGINEERING

Doctor of Philosophy

Engineering an Extensible Model for a Public Transport Journey Planning System

by Garrett Philip Fingerle

The research described in this thesis examined 1) the relevant issues of pre-trip public transport journey planning information; 2) the requirements for an extensible journey planning system; and 3) the necessary criteria from which to evaluate future, national, prototypes and models. It was found that, whilst comprehensive journey information is increasingly available via the Internet, it is not available in a coherent form and often requires the traveller to combine information from several sources in order to be able to decide on an integrated trip. The requirements necessary to allow a single entry point into a journey planning system were determined, a model developed, and a prototype of this model demonstrated using a protocol based on eXtended Markup Language (XML). In comparison to a warehouse model, a distributed data model and journey planning system design has the advantages of allowing expansion, both in geographical coverage and in flexibility of the protocol, in a relatively simple and incremental manner. Responses from individual components ranged from 0.05 seconds to 0.1 seconds for requests for points, and 0.1 and 0.8 seconds for the next journey, given a database measuring about 10 megabytes. Requests for the whole day ranged from 0.9 seconds to just under three seconds. Network overheads constituted one-tenth of these response times. Application processes enabling extensibility were an order of magnitude smaller than these network overheads. Trial journey requests with multiple modes, exchange points, and search engines provided response times under 30 seconds in serial operation, and less than half that in parallel. Thus overall response time of these trial journey requests is expected to be most affected by the serial or parallel operation of requests, the speed of the particular search engines, and the quantity of exchange points. Further progress will be a function of hardware, software, and network limitations, availability of more granular (i.e. intermediate stop point) schedule data, and improved data linkages through geographic relationships.
# Table of Contents

LIST OF TABLES................................................................. vi

LIST OF FIGURES............................................................. vii

PREFACE .................................................................................. viii

ACKNOWLEDGEMENTS............................................................. ix

ABBREVIATIONS....................................................................... x

DEFINITIONS............................................................................. xi

CHAPTER 1 - INTRODUCTION ...................................................... 1

1.1 PERSPECTIVE................................................................. 1

1.2 INFORMATION NEEDS OF THE TRAVELLER ...................... 2

1.3 OBSTACLES IN DISSEMINATION EFFORTS ......................... 3

1.4 REQUIREMENTS AND OBJECTIVES ................................... 3

1.5 APPROACH ........................................................................ 4

1.6 SUMMARY ......................................................................... 5

CHAPTER 2 - REVIEW OF PUBLIC TRANSPORT INFORMATION ......... 6

2.1 EUROPEAN RESEARCH, GEOGRAPHY AND REVENUE ........ 6

2.1.1 Relevant Research in Europe ........................................ 6

2.1.2 Political Structure and Names of Places in the U.K. ......... 7

2.1.3 Revenue Impact of Information ................................... 8

2.2 TRAVELLERS' NEEDS ..................................................... 9

2.2.1 Access to Information and its Interpretation ................. 10

2.2.2 Traveller Requirements .............................................. 11

2.2.3 System Requirements to Define and Plan a User's Journey 12

2.2.4 Importance of Response Time .................................... 13

2.3 EXISTING SERVICES ..................................................... 15

2.3.1 Historical provision of Public Transport Information .... 15

2.3.2 Traditional Publication Methods ................................. 17

2.3.3 Innovations in Information Provision ............................ 18

2.3.4 Combining innovations .............................................. 20

2.3.5 Application of innovations ........................................ 20

2.3.6 Public Transport Resources on the Internet ................. 21

2.3.7 Summary of Available Information Services ................ 23

2.4 EXISTING SOURCES OF JOURNEY PLANNING DATA ....... 25

2.4.1 Data Requirements for Journey Planning .................... 25

2.4.2 Names of Places Data ............................................... 27

2.4.3 Exchange Point Data ................................................. 28

2.4.4 Scheduled Service or Trip Data ................................. 29

2.5 METHODS OF DATA TRANSFER .................................... 29

2.5.1 History of Data Dictionaries ...................................... 29

2.5.2 Analysis ................................................................. 31

2.5.3 Terminology ............................................................ 32

2.6 SURVEY OF JOURNEY PLANNING RESOURCES ............. 34

2.6.1 Functions of Electronic Journey Planners .................... 34

2.6.2 Components of a Journey Planner .............................. 35

2.6.3 Criteria used for Journey Planning ............................. 36

2.6.4 Hardware and Software in surveyed systems ............... 38

2.6.5 Enquiry and Search Engines ..................................... 38

2.6.6 Presentation of Information ....................................... 39

2.7 SUMMARY ..................................................................... 40

CHAPTER 3 – REVIEW OF SYSTEM ARCHITECTURES ..................... 42

3.1 BACKGROUND IN DATABASE TECHNOLOGY ..................... 42

3.2 BASIC ARCHITECTURES ................................................. 43

3.3 DESIGN, ANALYSIS AND EVALUATION ......................... 45
3.3.1 Design Parameters .................................................. 45
3.3.2 Data Maintenance, Data Accuracy, and Integrity ............... 46
3.3.3 Security ............................................................... 47
3.3.4 Extensibility .......................................................... 47
3.3.5 Redundancy ........................................................... 48
3.4 Influence of Engine and Transmission Speed on Design ....... 48
3.4.1 Theoretical Internet Speed ....................................... 49
3.4.2 Theoretical Engine Speed ......................................... 50
3.4.3 Design Decisions .................................................... 51
3.5 Specific Architectures for Multiple Databases .................... 52
3.5.1 Alternative One ..................................................... 52
3.5.2 Alternative Two ..................................................... 52
3.5.3 Alternative Three ................................................... 53
3.6 Analysis of Alternatives: Chosen Design ......................... 54
3.6.1 Flow Diagram ....................................................... 55
3.7 Summary ................................................................... 57

CHAPTER 4 – REVIEW OF DATA MESSAGING .......................... 59
4.1 Background of Data Exchange ....................................... 59
4.1.1 Data transport and the Internet .................................. 59
4.1.2 Background on Capacity ........................................... 60
4.1.3 Quantity of Data for Transmission .............................. 60
4.1.4 Analysis of Internet protocols ..................................... 61
4.1.5 TCP/IP based methods of Data Exchange .................... 62
4.2 Existing Message Standards ........................................ 64
4.2.1 Header of Group Message ........................................ 65
4.2.2 Body .................................................................. 65
4.2.3 Footer of Group Message .......................................... 65
4.3 Obstacles Preventing Data Exchange .............................. 66
4.3.1 Different Hardware and Operating systems .................. 66
4.3.2 Different naming conventions .................................... 66
4.3.3 Different coverage in services .................................... 67
4.3.4 Different methods of data storage .............................. 67
4.3.5 Different variable names .......................................... 68
4.4 Message Format ......................................................... 69
4.5 Summary ................................................................... 71

CHAPTER 5 – REVIEW OF METHODOLOGIES ....................... 73
5.1 Introduction ............................................................... 73
5.1.1 McCormack’s methods ............................................. 73
5.1.2 Industrial methods .................................................. 74
5.2 Journey Planning ........................................................ 75
5.3 Issues in Creating the Journey Definition ........................ 76
5.3.1 Obstacles in defining a date or time ............................ 77
5.3.2 Obstacles in defining an origin or destination ............... 78
5.3.3 Maintaining a record of past user-decisions ................. 79
5.3.4 Mitigation of Issues in Name Definition ..................... 81
5.3.5 Summary ............................................................. 81
5.4 Obstacles in Determining an Itinerary .............................. 83
5.4.1 Journey Types and Routing ....................................... 83
5.4.2 Data Precision ........................................................ 85
5.4.3 Remote Journeys ..................................................... 86
5.4.4 Exchange Points between Transport Networks ............ 87
5.4.5 Methodology for POST Requests ............................. 88
5.4.6 Routing Field Structure within POST method ............. 89
5.4.7 Summary ............................................................. 89
5.5 Generic Approach ....................................................... 90
5.5.1 Defining the Journey ............................................... 90
5.5.2 Finding Solutions to the Journey ............................... 91
5.5.3 Testing ............................................................... 93
List of Tables

Table 1: Harris survey of public transport information ......................................................... 11
Table 2: Summary of surveyed planning criteria: Input .......................................................... 12
Table 3: Summary of surveyed planning criteria: Output ....................................................... 13
Table 4: Sources of public transport information ................................................................... 17
Table 5: Statistics relating to PLACES .................................................................................... 27
Table 6: Statistics relating to EXCHANGES ........................................................................... 28
Table 7: Design parameters for a National Public Transport Information System .................. 29
Table 8: List of Data definitions for ATCO.CIF ....................................................................... 32
Table 9: Transit Communications Protocol definitions (ST-ITS-TCIP-SCH, Feb 1998) ........... 34
Table 10: a listing of most Journey Planner Suppliers and their Products ............................... 39
Table 11: Examples of different naming for a point ................................................................. 66
Table 12: Keywords used for 'origin' and 'destination' ......................................................... 68
Table 13: Several Date and Time formats .............................................................................. 69
Table 14: Test Apparatus for an Internet Echo client and server ............................................ 97
Table 15: Test Apparatus for an Internet WWW client and server ......................................... 97
Table 16: Test Apparatus for an Internet WWW client and server ......................................... 100
Table 17: Functions of tested systems .................................................................................... 112
Table 18: Data Relationships ................................................................................................. 115
Table 19: Tests showing the Relative Impact on Performance ................................................. 117
Table 20: Test to show merits of a local or centralised National Gazetteer ............................. 119
Table 21: Sample Tabulation Comparing effects of Assembly and Complexity on Response Time ...... 122
Table 22: Histograms at different measurement points for a PIII-500 with 50 iterations .......... 130
Table 23: Histograms at different measurement points for a PIII-500 with 500 iterations ....... 130
Table 24: Source Data Set One - 181 - Pentium II - 450 ...................................................... 134
Table 25: Source Data Set Two - 181 - Pentium II - 450 ...................................................... 134
Table 26: Data set one - Exchange - Pentium III - 500; 500 samples ........................................ 134
Table 27: Data set two - Exchange - Pentium III - 500; 500 samples ....................................... 134
Table 28: Origin Destination Pairs for Search Request Tests .................................................. 137
Table 29: Measured time for the engine to respond to enquiry request ................................... 137
Table 30: Estimated response time of the routing engine ....................................................... 138
Table 31: Engine and Network response time for "interval" equals 20 hours ............................ 140
Table 32: Approximate ratio comparing the time to retrieve a whole day's journeys verses just the next few .................................................................................................................... 141
Table 33: Time to write and read "swap" file and time read input from incoming request stream ...... 144
Table 34: Response times for individual tasks within journey planning .................................. 146
List of Figures

Figure 1: Detail of a Journey using two Services ................................................................. 26
Figure 2: Journey Planner Components ................................................................................. 35
Figure 3: Warehouse Design Alternative ............................................................................... 52
Figure 4: Single Broker Design Alternative .............................................................................. 53
Figure 5: Generalised Public Transport Information System .................................................. 54
Figure 6: A model of a Journey Planning System ..................................................................... 55
Figure 7: Flow diagram of data exchange between components of the journey planning system .................................................................................................................. 56
Figure 8: Sample Network ..................................................................................................... 60
Figure 9: Sample bus timetable ............................................................................................... 61
Figure 10: Group Message Structure ....................................................................................... 64
Figure 11: Message Structure ................................................................................................ 65
Figure 12: Element Structure .................................................................................................. 65
Figure 13: Sample XML show service information ................................................................. 71
Figure 14: Translation of Operator Names to Common Name .................................................. 78
Figure 15: Sample web page refining the origin within Buckinghamshire ............................... 80
Figure 16: Route/Service between Area A and Area B .............................................................. 84
Figure 17: Service joining two non-adjacent political areas in two adjacent database areas ........................................................................................................................................... 84
Figure 18: Staffordshire and Shropshire county areas ........................................................... 86
Figure 19: three stage "cross boundary" journey – one software supplier, two modes .......... 93
Figure 20: three leg "cross boundary" network – multiple software suppliers, modes, and exchange points .................................................................................................................................................................................................................. 94
Figure 21: The Heterogeneous Test System .......................................................................... 101
Figure 22: Sample image showing Initial Point of Entry screen ............................................. 107
Figure 23: Test Apparatus ...................................................................................................... 127
Figure 24: Mean Response Time for Place command with 500 samples ................................ 132
Figure 25: Relative size of Engine and Network Time for Place command ........................... 133
Figure 26: Relative Engine and Network Mean Time of 50 samples on a Pentium III - 500 .... 139
Preface

At least I have greater confidence in travelling by public transport now ... may that others benefit from this knowledge too
Acknowledgements

In memory of my father and great aunt, dedicated to my wife and son, and with thanks to my sisters, brothers, mum and dad.

I thank foremost my wife, Vanessa, for her intellectual and emotional support and encouragement, which helped me to see this goal through to fruition. I also thank my parents for their considerable support. It is necessary to also thank, in particular (in alphabetic order) David Dyson, Paul Houghton, Dr. Anthony Lock, Professor Mike McDonald, Mike Ness, and Roger Slevin for their considerable help, support, and encouragement. I wish also to acknowledge specific efforts by Steve Alexander, Bill Banks, Peter Bates, Bob Bourne, Chris Brown, Tony Brown, Simon Day, David Jeffery, Tony Ferguson, Chris Gibbard, John Harris, David Jones, Brian Marshall, Gordon Moore, Peter Stoner, Howard Wyborn, and various others for their efforts and more over, their patience and understanding.
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EC</td>
<td>European Commission</td>
</tr>
<tr>
<td>EU</td>
<td>European Union</td>
</tr>
<tr>
<td>GB</td>
<td>Great Britain</td>
</tr>
<tr>
<td>UK</td>
<td>United Kingdom</td>
</tr>
<tr>
<td>USA</td>
<td>United States of America</td>
</tr>
<tr>
<td>DETR</td>
<td>Department of the Environment, Transport, and the Regions (GB)</td>
</tr>
<tr>
<td>(US)DOT</td>
<td>United States Department of Transport</td>
</tr>
<tr>
<td>DTI</td>
<td>Department of Trade and Industry (GB)</td>
</tr>
<tr>
<td>ANSI</td>
<td>American National Standards Institute</td>
</tr>
<tr>
<td>BSI</td>
<td>British Standards Institution</td>
</tr>
<tr>
<td>CEN</td>
<td>Comité Européen de Normalisation</td>
</tr>
<tr>
<td>EDIFACT</td>
<td>Electronic Data Interchange for Administration, Commerce and Transport</td>
</tr>
<tr>
<td>EBES</td>
<td>European Board for EDI Standardisation</td>
</tr>
<tr>
<td>GTDI</td>
<td>Guidelines for Trade Data Interchange</td>
</tr>
<tr>
<td>IEEE</td>
<td>Institute of Electrical and Electronics Engineers</td>
</tr>
<tr>
<td>ISO</td>
<td>International Organisation for Standardisation</td>
</tr>
<tr>
<td>UNTDID</td>
<td>United Nations Trade Data Interchange Directory</td>
</tr>
<tr>
<td>IDE</td>
<td>Integrated Drive Electronics</td>
</tr>
<tr>
<td>RAID</td>
<td>Redundant array of inexpensive disks</td>
</tr>
<tr>
<td>SCSI</td>
<td>Small Computer System Interface</td>
</tr>
<tr>
<td>OSI</td>
<td>Open Systems Interconnection</td>
</tr>
<tr>
<td>TCP/IP</td>
<td>Transfer Control Protocol/Internet Protocol</td>
</tr>
<tr>
<td>UDP</td>
<td>User Datagram Protocol</td>
</tr>
<tr>
<td>ADSL</td>
<td>Asymmetric Digital Subscriber Line</td>
</tr>
<tr>
<td>ISDN</td>
<td>Integrated Services Digital Network</td>
</tr>
<tr>
<td>LAN</td>
<td>Local Area Network</td>
</tr>
<tr>
<td>MAN</td>
<td>Metropolitan Area Network</td>
</tr>
<tr>
<td>PSTN</td>
<td>Public Switched Telephone Network</td>
</tr>
<tr>
<td>PSDN</td>
<td>Public Switched Data Network</td>
</tr>
<tr>
<td>WAN</td>
<td>Wide Area Network</td>
</tr>
<tr>
<td>ASP</td>
<td>Active Server Pages</td>
</tr>
<tr>
<td>CGI</td>
<td>Common Gateway Interface</td>
</tr>
<tr>
<td>FTP</td>
<td>File Transfer Protocol</td>
</tr>
<tr>
<td>HTML</td>
<td>Hyper Text Markup Language</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hyper Text Transfer Protocol</td>
</tr>
<tr>
<td>ISAPI</td>
<td>Internet Server Application Programming Interface</td>
</tr>
<tr>
<td>RFC</td>
<td>Request for Comments</td>
</tr>
<tr>
<td>SMTP</td>
<td>Simple Mail Transfer Protocol</td>
</tr>
<tr>
<td>SNMP</td>
<td>Simple Network Management Protocol</td>
</tr>
<tr>
<td>XML</td>
<td>eXtended Markup Language</td>
</tr>
<tr>
<td>XSL</td>
<td>eXtended Style Language</td>
</tr>
<tr>
<td>WWW</td>
<td>World Wide Web</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATOC</td>
<td>Association of Train Operating Companies</td>
</tr>
<tr>
<td>ATCO</td>
<td>Association of Transport Co-ordination Officers</td>
</tr>
<tr>
<td>CCITT</td>
<td>International Telephone Telegraph Consultative Committee</td>
</tr>
<tr>
<td>PTI</td>
<td>Public Transport Information</td>
</tr>
<tr>
<td>TAN</td>
<td>Traffic Area Network</td>
</tr>
<tr>
<td>TC</td>
<td>Traffic Commissioner</td>
</tr>
<tr>
<td>TT&amp;L</td>
<td>Travel, Tourism and Leisure</td>
</tr>
</tbody>
</table>
Definitions

In this thesis, the following have been taken to have these specific meanings:

**Point:** a geographical location in physical space given to the nearest metre. A point may be on the ground (a bus stop), above ground (a gate in an aircraft terminal wing or satellite), or below ground (a bay of a metro platform). It need not be a point on the transportation network (i.e., it can be the entrance to a house, abode, office, or "point of interest"). Note: the issue of accuracy is important in so far as one must be able to determine one bus stop (pole), train platform (bay), or physical street address from another.

**Route:** a collection of linked points, along connecting streets or fixed guide-way networks (i.e., a railway line)

**Service:** a specific collection of points along a route, by a specific operator.

**Stop:** a *point* along a transport network where a traveller may depart (only), alight (only), or depart and alight from a *service*.

**Cluster:** an ad-hoc collection of *stops*.

**Mode:** a description of the method of traversal between two points in space whether in a vehicle or not; i.e., walking, cycling, taxi, ferry, rail, bus, aeroplane, car, etc.

**Trip (leg):** a movement between two points by one mode (sometimes also referred to as a "trip segment").

**Journey:** the movement between the point of origin and the point of destination. Normally there are at least one exchange point and at least two trips (legs/segments). It is noted that some journeys may have only one trip (i.e., a cycle or walk trip), and no exchange points.
Chapter 1 - Introduction

1.1 Perspective

Population in Great Britain has grown by 20% within the last six decades, reaching nearly 60 million people. However, mobility has seen a 400% growth over the same period to 720 billion passenger kilometres per year. Over 100% of this growth can be wholly attributed to increased private car and aircraft use (DETRf, 2000 and ONS, 2000). As over 95% of this mechanically assisted transport, both in the UK and globally, is fuelled by oil (Johansson, 1996), this growth in passenger kilometres may explain two facts. First, the transport sector has increased its percentage of total energy consumption from 15% to over 30% in the last six decades. Second, gross energy consumption has also increased, and with this increase, gross oil combustion also has increased (EuroStat, 1996, 1999). Through this increase in combustion, the relative atmospheric quantities of water vapour ($H_2O$), sulphur oxides ($SO_x$), nitrogen oxides ($NO_x$), and carbon oxides ($CO_x$) have increased in many local areas and globally. Adverse effects often result, coupled with increases in noise from this four fold increase in car and air traffic (DETRc, 1998). Another concern is that the natural production rate of petroleum, gas and coal stocks suggest a finite limit in the practical availability of these fuels, whether for households, commerce or transport, occurring within the next century (Cunningham, 1990).

The above trends highlight the need for more efficient use of transport infrastructure and possibly a significant decrease in mobility of particular social groups also. Realising the importance of these trends, governments began a series of commitments to reduce pollution, and encourage integrated and sustainable transport (DETRc, 1998). One method of realising these commitments is to reduce society's dependence on the private car and air transport. Three options of reducing this dependence include 1) the restriction of these modes, 2) the promotion of alternatives such as walking, cycling, and 3) increased utilisation and provision of public transport, including bus, coach, and rail (DETRc, 1998).

A number of barriers exist to increasing utilisation of existing services. Existing services often lack priority, new vehicles, an adequate supply of drivers, and information for passengers regarding services (Morris, 2000). Where such information exists, Balcombe (p6, 1997) states that passengers have a lack awareness of the service information. Further, Balcombe (p24, 1997) and Wilkinson (p 28, 1998) claim that "most passengers" do not need information. However, interpretation of their research suggests that "passengers" are already familiar with services and do not have need of any further information, unless the services change. Due to bus and coach deregulation, services change more frequently (Capes, 1991), and thus the percentage of journeys which are "new" has increased. As 80% of users need information for new journeys, (Balcombe p11), this suggests that information could help increase utilisation in a deregulated bus and coach market. Whilst information provision about rail services is more centralised than bus and other services, integration with bus or other modes is still an unresolved issue (Wilkinson, 30, 1998).
Barriers also exist to increased provision of new services. The provision of new services requires new funding. If better utilisation of existing funding goes to shareholders, another source of funding is necessary. One such source of funding would be from new users, but to attract new users, someone needs to provide information on the new services. Furthermore, Balcombe (p9-11) suggests that the lack of service information can act as a deterrent to use. Their research suggests that users without service information were deterred from travelling in a quarter of regular journeys, one third of occasional journeys, and over three-quarters of new journeys. However, research by Transport for London (Le Juene, 1992) suggested that investing in information could make a small increase in revenue. If this is the case, then information could help to fund new services. However, outside Greater London, increased revenue may simply go straight to shareholders. These arguments suggest that information provision should involve government. Research by the DETR (a, c, d, g 1996 - 2000) suggests that potential travellers or travellers\(^1\) need information for all services to help encourage the use of public transport and thus reduce the environmental consequences travel.

1.2 Information Needs of the Traveller

Given this impetus and focusing on new journeys, the information needs of the user fall into four fundamental categories (Bernard, 1995), (Casey et al, 1998), (Huske, 1999), (Infopolis, 1999), (Kabjorn, 1999), (Tongeren, 1999). Firstly, users need accurate routes, times or schedules, and prices of available services between two points (i.e. static information). For indirect journeys, users need to know the combination of services that would allow a complex journey, with intermediate exchange or transfer points at which the individual needs to know precisely how to transfer between services, of the same or a different mode. This category is termed itinerary, trip or journey planning information, and is usually necessary before the user embarks on their journey. This informational need also applies to users whose regular service(s) have experienced a change. Thirdly, users, particularly if they are unfamiliar with the journey, seek to know the status of current and subsequent services, termed real-time information. Finally, travellers may want to know what to do should something go wrong whilst en-route, or just before they depart. This category is contingency information. Wilkinson, (pg 25, 1998) refers to this as dynamic information.

Within the more limited scope of static and journey planning information, work by Infopolis (1999) and Heym (1999) state the obvious, i.e. that individuals make different types of journeys. Individuals may be considered to start a 'journey' at a location. They go to one or more locations, spend some time at any one location along the way, such as an office or store, and, eventually, return to the original location. These journeys are termed return or round-trip journeys. If they go via one or more intermediate locations before returning to the original location, it is sometimes called an open jaw. Alternatively, their journey may only be one way or single. Journeys also vary in length. Most (i.e. 80%) journeys are local, with a much smaller percentage that are "mid distance" in length. Finally, perhaps another order of magnitude smaller is the long distance journeys. (Wilkinson, pg 43, 1998)

\(^1\) often collectively referred to simply as "users" or "end-users"
Any of these types of journey may utilise **multiple modes**. Three examples include 1) walk a short distance and take the bus, then walk; 2) walk to the station, catch a coach, then walk to a connecting train or aeroplane, then walk and catch a cab; or 3) cycle to the station, catch a train and then walk to the rank for a taxi. The journey may also involve a number of different transport service companies, even for a short, direct, return journey. For instance, they may take a bus operated by one company on the outbound trip, and the same type of bus but operated by a different company on the inbound trip.

### 1.3 Obstacles in Dissemination Efforts

Available schedule information is not so flexible. Information may be hard to read, difficult to decipher, limited to one service operator or mode, incorrect (perhaps inaccurate and imprecise), missing beyond some vaguely defined boundary, out of date, or missing altogether (DETRA,c 1996 1999) (Tyler b, 1999). To achieve a "seamless" journey, the individual would need comprehensive journey planning information that overcomes these inadequacies and other obstacles.

The distributed nature of the public transport network is also an obstacle in the UK and elsewhere. Perhaps as a result, the existing supply of journey information in Britain contains a heterogeneous mix of databases, search engines and user interfaces by competing suppliers, as surveyed in Chapter Two. These databases have political and geographic boundaries, and most of these systems contain some element of overlay with neighbouring systems. Furthermore, systems have to cater for different currencies, multiple languages and numerous names for the same location. (Ferguson, 1999), (Slevin, 1996), (Trongeren, 1999), (Wheeler, 1999), (Wilkinson, pg 39-40, 1998)

A number of other obstacles exist. Two examples of bias include promoting one operator through the manner in which the information is presented, and through the computation processes of calculating journeys (Brown, 1998). Finally, limited budgets constrain the ability to disseminate the information, and potential response speed of the system has to be quick. (CETE, 1995), (DETRA, 1996), (Kabjorn, 1999), (Paylor, 1999), (Slevin, 1996).

To overcome the above obstacles without computerised assistance in a privatised, deregulated environment is prone to error and high costs. If a computer system were to automate the provision of information for wide-scale use, it could aim to have a transparent cost to the 'user' and give an unbiased, individualised answer from anywhere (i.e. "a one stop shop", Prescott, 1999) to a personalised enquiry and response. No system yet meets these aims.

### 1.4 Requirements and Objectives

The introductory paragraphs suggest a number of obstacles preventing comprehensive fulfilment of users' needs. The research for this thesis concentrates on the necessary functions to achieve an extensible
journey planner model. It was necessary to further constrain the research by selecting six specific functions that overcome particular obstacles described in the previous section. These functions might be summarised as follows:

- Ability to function across political boundaries
- Ability to expand journey planning to include other modes
- Ability to allow journey planners to interrogate other journey planners
- Ability to allow journey planning to function with areas missing
- Ability to function independent of hardware and software platforms
- Ability to function independent of private database protocols

From these desirable functions, it was possible to suggest specific objectives for this research. It is noted that the research began in 1994, and focused on the emerging issues that have become more important recently as a result of research by the DETR (a, c, d, g 1996 - 2000).

The aim of this research is to examine potential methods which could enable public transport journey planners to provide an integrated, impartial journey itinerary, crossing administrative or database boundaries, and may include more than one computer platform, mode and industrial supplier, in a resilient and efficient manner. In particular, it tests response times at certain points within a laboratory-based prototype system, and examines the potential consequences of various design decisions in detail. Thus a key objective is to provide an indication of which fundamental decisions would critically influence a 'timely' response, and postulate possible consequences. However, the tests conducted in this thesis specifically excluded the effects of network communications passing through multiple routers and over long distances as may occur should a real implementation of the methods in this research be undertaken. The self-contained laboratory setting will enable isolation of indicative factors affecting extensibility within an internal network, in conjunction with "remote" data access of public transport information via a search engine.

1.5 Approach

Research studies normally commence with a literature review, and follow on with a review of objectives and further examination of key processes. Most studies also involve modelling or building a prototype and evaluating it. The research for this thesis varied this approach slightly, as four separate, but related reviews of current knowledge were undertaken largely in parallel.

Chapter Two examines public transport information systems in greater detail, and the different terminology used. The activity determines a common, perhaps minimal, set of parameters, to describe salient features to define a 'trip', regardless of mode. This chapter also looks at the quantities of data and their sources in providing traveller information, and at existing methods of data exchange. Chapter Three examines databases, data storage, and computer networks. Chapter Four reviews the structure of
messages and their formats, whilst Chapter Five reviews what is later described as gazetteering and routeing. Chapter Six describes the apparatuses built for testing 'proof of concepts', demonstrating and prototyping how the concepts, which form a significant contribution of this work, fit together. Chapter Seven tests a number of aspects of a partial prototype system and critically evaluates these tests, leading to suggestions for further work. Due to the pace of change in this research area, the 'literature review' has had to be ongoing through the study. As a consequence, some of the material that makes up initial chapters has only recently been published. Thus, whilst a concerted effort has been made to include key recent work (i.e. published between late 1997 and 2001) in the initial chapters where relevant, not all related research may have been included.

1.6 Summary

Trends suggest that computer technology may be sufficiently efficient in the near future to enable an extensible Public Transport Information (PTI) system. This PTI system could be capable of providing a selection of alternative, multiple mode, journey itineraries to a multitude of potential travellers. Any single traveller with a desire to travel from any point in one country to another point in the same or another country using only public transport (i.e. plane, train, metro, coach, bus, taxi, or walking, and cycles) could use the system. The focus of this research is on how a system of computers and journey planners could be designed, engineered and interconnected to supply a low cost, extensible, public transport journey planning service which is free to the user. In particular, it tests a partial prototype to determine critical aspects of a potential system, postulating potential effects on response time of any indicators.
Chapter 2  Review of Public Transport Information

2.1 European Research, Geography and Revenue

The first part of this Chapter examines relevant research elsewhere, noting that much of the research reviewed was published in parallel with, or after the research for this thesis was completed. It also examines the geographic and political origins of numerous problems that arise by focusing on just the United Kingdom, even though aspirations held a wider geographic context. Subsequent paragraphs review studies into the impact of specific systems and, particularly, the potential for revenue generation. Sections discussing the information requirements of the traveller, available information services, available data, and methods of data transfer, follow this review of revenue generation.

2.1.1 Relevant Research in Europe

EUROTRIP (Grill, 1990) was the first major attempt at an integrated public transport information system in Europe. The EUROTRIP project concluded that the technology existed, but it was neither 'cost effective' nor widely available. The lack of cost effectiveness could be interpreted as not finding a profitable business model; i.e. travellers were not likely to pay enough for the information to make the system viable. Computer infrastructure that is increasingly available in government and company offices in 2000 was not affordable in 1990 and thus not widely available. It is relevant to note that research by Balcombe (1998) suggested that users were unlikely to pay towards such a system, even towards the end of the decade when improved computer hardware was widely available.

In the last decade, the research from several studies described below might be grouped into two areas. One area might be termed a 'user studies' area, which tries to model and improve interactions between the 'user' and the 'system'. A second area is primarily concerned with interactions between components within the 'system'; sometimes termed the 'back-office'. In the research studies examined, but completed previous to 1998, significant overlap existed. More recent research studies have become more focused on one of the two areas.

Research in the 'user studies' area includes complementary projects of INFOPOLIS I and II (1999), EUROSCOPE (1999), and VADE MECUM. Their aims centred on making recommendations for user interaction with the system and presentation of information from the system to the user. Behavioural techniques and limited prototypes were used as numerous obstacles prevented the application of systems such as that which formed the basis for this thesis. Other European Commission Fourth Framework programmes from 1994 to 1998 included projects such as TURTLE (1994-8), PROMISE (Tate, 1995), and TABASCO (Catling, 1996), which developed or enhanced existing kiosk-based journey planners or similar prototypes as part of their research programmes. The TITAN (CETE, 1999) project further
developed a standard data dictionary, EUROBUS (CETE, 1995, 1999), which was limited to bus transport. The VITAL (Visitors' Information on Transport and Locations, Amsterdam and Brussels) project developed a segregated bus, car or walk journey planning system for visitors to city centres. Other projects with European funding included TRIP (Esprit programme), INFOTEN (Tognoni, 1996), ROMANSE (Mansfield, 1995), and SCOPE (Stephanedes, 1993). MATIS (Mouskos, 1995) is a project from the USA. Later systems included limited World Wide Web prototypes.

The Dutch national railways, with several partners, designed a journey planner system to support their telephone enquiry centres, named OV Reisplanner (OVR, 1994). It integrates the bus and rail modes, and provides integrated times, routes, and fares. This consortium (Potgraven, 1996) used a distributed file design so as to minimise response time for its call centre environment (Boot, 1994). However, their journey planner is limited geographically to the Netherlands and to one software supplier for the consortium. The prototype system reviewed for this research did not include air or ferry modes. Finally, the Dutch system did not anticipate the need to exchange data externally.

Demonstration software by Duetschebahn (DB) concentrated on heavy rail and light rail, but has recently been expanded to include other ground modes through the EU-SPRIT project (Brigge, 1998-9). Related work includes EFA and DELFI, the latter of which was funded by DB. Current literature by the EU-Spirit project suggests limitations in extensibility (walk, bus, air and ferry) and overall reliability of the integrated software system. TRIDENT (Booth, J 2000) and PEPTRANS (Wheeler, 2000) are Fifth Framework projects which may build on previous work.

2.1.2 Political Structure and Names of Places in the U.K.

The United Kingdom offered an excellent representative basis for a study of this nature, as it is diverse in political structure and languages. At the 'national' level, the United Kingdom has four quasi-independent "states", i.e. England, Wales, Scotland, and Northern Ireland. Whilst English is the common language, each state has its own dialect or even language, which results in the same place being called different names (i.e. Swansea). Below the national level, each 'state' has counties or local authorities within it. Urban or Metropolitan areas may relate to specific local authorities. Legal powers vary. Local authorities sometimes are the principle result of the "town", which historically evolved as a result of "charters". Towns may have "wards" as a political sub-unit. Localities typically reside within a "parish", or what is called a "community" in Wales. These political structures do not have an absolute hierarchy, i.e. some interleaving occurs. Coupling this with the 1998 reorganisation of local government has resulted in a political mélange.

This diversity in government structures has resulted in a lack of uniformity for tendering transport services in a partially privatised and deregulated market. London has a great deal of regulation, but suffers from disintegration between modes. In Manchester, competing services vie for passengers on the same route at the same time. In the 'shire' counties, the local governing bodies may tender some services.
that are not commercially viable but socially needed. Services that are commercially viable may also be under tender in some areas, but given the same situation elsewhere, no tender is necessary. Thus competing services in the same market sometimes provide what is essentially a single service to the user but may employ minor differences in brand or otherwise to distinguish themselves. Ironically, this mélange is partly due to EU legislation attempting to harmonise practices within its member states.

This seemingly ad-hoc local implementation of policy is further complicated by the manner in which bus operators register the details of their services with the regional Highways Agency. The details of the route and the service number are set down in the "registration". These details vary substantially in accuracy and format. For instance, they may include geographic co-ordinates cross-referenced with a table of names and aliases. Or they might simply be a line drawn on a paper map that is out of date. Sometimes material includes references to colloquial sources, and thus is ambiguous in a national context.

As a result of the variance in the naming details used in the registration process, a subsequent project completed between February and November, 2000 called TransXchange (DETRg, 2000) was commissioned by DETR and TAN hoping to achieve some commonality for bus transport. Similar scenarios exist for other modes of transport also. (DETRa, c, g, 1996, 1998, 2000)

An obvious requirement for a human or computer-based journey planning system is get the traveller to decide on an origin and a destination that is correctly understood by the 'system'. The seemingly trivial nature of this masks its complexity. As an example of the difficulties that can occur in practice, a traveller may choose to name a place using one or more names related to political hierarchies discussed above. Being local, the traveller could simply state the name of the road or local building, or the name of the stop as it is stated on the pole or shelter at the side of the road, and thus the town name is implicitly understood (by the driver). The traveller may know this stop by what it used to be called. Thus it may not be the name of the current building, current road name, or the current stop name. (Wilkinson, 1998)

Given a profile of many bus users based purely on observation (under 18, over 60, disadvantaged, female, foreigners or tourists), a wide variety of names can be expected for a particular location. Rumour has it that in some areas, examples still exist where the road sign has not yet been replaced since removal during the last World War.

These examples clearly show a difference in the way public transport operators, particularly bus operators, describe their services and how the public names a location. The problems are further exasperated by a lack of logical relationships between existing data sets (see section 2.3). This chasm poses a significant challenge for the provision of integrated information on public transport services.

2.1.3 Revenue Impact of Information

Discussion in the Introduction suggested that much of the potential revenue from a journey planner system might go to the shareholders in a privatised market. However, as a grocery store does not charge the shopper to know what goods it displays and sells on its shelves, it is considered that the expectation of
the traveller will be that information should be given freely to the traveller. Research on "user requirements" also suggested this (Lyons, 7.4.1, 1999). Given sufficient collaboration and collective effort, this thesis assumes that a group of computers could calculate a "reasonable" selection of alternatives sufficiently quickly and accurately even if the task occurs in excess of a million times a year (Harris, 2000), and require significantly less human effort.

In a quasi-public market situation, Loop et al (1994) concluded that the introduction of improved traveller information had a significant and measurable effect, and could be self-sufficient financially given certain pre-conditions. Such conditions included a reliable system, and response times that were sufficiently quick. Given these conditions, Potgraven (1996) and (Laconte, 2000) suggested that increased public transport information systems could be instruments for successfully implementing transport policies more geared towards reducing car dependence. In a study for London Transport, Le Jeune (1992) suggested that revenue was nearly five times the expenditure on telephone enquiry systems.

A subsequent research study by Harris (2000) sought to determine any likely benefit from "trip generation". The study defined two types of enquiries: generative (+) & abstractive (-). Their definitions follow:

**Generative**
- person switched from all private to all public transport
- from no trip to public transport
- from lower fare to higher fare using public transport

**Abstractive**
- switch from public to private transport
- from planned transport to no transport as a result of information
- from higher fare to lower fare as a result of information

Three other outcomes were measured also. These outcomes were 1) neutral, 2) "depending", and 3) unsure or incomplete outcome. Their research, as did previous studies (NRES, 1999; TfL, 1997—see Harris (2000) for details), suggested that less than 5% of respondents enquiries led to generative trip generation based on the availability of information. These results indicate that any potential benefits for revenue creation from the research described in this thesis may be limited. As the impetus of this research is environmental, it nonetheless investigates the internal processes of data interaction between computers within national/global system to increase the collective impact. The potential for improved information would help many people whom are dependent on public transport, and instil confidence in making the journey with departure and arrival times, and the location of any connecting services and other relevant information.

### 2.2 Travellers' Needs

Having discussed a broader background with respect to existing research outside the UK, geographic issues, and the potential for revenue generation, the following paragraphs focus on the traveller's needs, and how existing work has tried to meet those needs.
2.2.1 Access to Information and its Interpretation

Research commissioned by Railtrack (1998) suggested that "...access to information about alternative modes is a problem, though new technology can help. More than three out of five people do not feel well informed about local bus and train routes and timetables, or about local cycle routes. Three out of four do not feel well informed about domestic flight routes and timetables."

"Yet most would be inclined to use some other modes more if they were better informed. If they had better knowledge of routes and timetables, four out of ten said they would make more use of local buses, one out of four local trains, one in five long distance buses and one in seven long distance trains. One in seven also said they would use local cycle routes more if they had better knowledge of where they were."

"Often it is lack of effort on the part of the public, but service providers do not necessarily make it as easy as they could for the public to know what services are available. Most people consider it easy to get hold of information on local bus services and on local train services."

"They are more reserved about how easy it is to find out information about long distance bus, and rail services and domestic flights, and they are highly likely to consider it difficult to be able find out about local cycle routes. Current methods people use to get information include "word of mouth", maps, timetables or schedules, fares tables, itinerary planners, and real-time displays."

Balcombe (p6-7, 1997) investigates the issues of access and interpretation in greater detail. Their research of Hertfordshire, North Yorkshire, West Midlands, and Greater Manchester suggested that existing passengers had a relative satisfaction level of about three on a scale of zero to five on aspects such as information, frequency, and reliability. Their work also found considerable regional variations, with users in Manchester having twice as much awareness of services as users in Hertfordshire.

Where a service had a high frequency, Balcombe (p8, 1997) found that most people did not bother using the timetable. The percentage of passengers consulting timetables increased linearly with a decrease in service frequency, suggesting that 40% of regular and occasional passengers consult timetables on half-hourly services. Once the service frequency descends below every half-hour, the number of passengers consulting timetables increases more rapidly. Balcombe (p9, 1997) also discussed a number of problems with passengers being misled and perceiving that information is inaccurate.

Tyler (b, 1999) points out that the print is often too small with available information, even if the user had fully corrected vision. Also, where stops are unlit, reading existing timetables at the stop is further impeded before sunrise and during/after sunset. Depending on the height of the passenger, he or she may need to stoop or look up to see the small print.

Research by Harris (2000) into a proposed National Public Transport Service attempted to summarise the relative importance of traveller information on decision-making. The following table summarises the questions and responses briefly.
Service description: "In addition to enquiry lines and websites which currently exist, a new national public transport service may be introduced. You could use one telephone number for any local or national public transport information. For instance, you could simply ask how to get from one town or village to another town or village, and you would be told which methods of transport were available and you would be offered route and timetable information. Calls would be charged at nation rate, which would not be more than 8 pence per minute." (Harris, 2000)

Survey Question | Response (Range)
--- | ---
How many calls will be made? | Between 26 and 60 million per year
Who are the potential users? | Current private and public transport users
What is the relevant importance of the availability of information when considering whether to use public transport? | 1% mentioned better information (unprompted)
 | 36% mentioned better information (prompted).
How easy is it to get information? | 75% of bus and rail users think it is easy to find information about services
Would more information increase patronage? | 15% probably, 25% perhaps, and of these 40%, when asked why they did not have it, 60% could not bother to look, 30% did not know where to look, and only 10% tried (all failed)
Would respondents use the service? | 15% no, 25% not likely, 30% likely, 30% v. likely
the "likely" and "very likely" had a significantly higher percentage of public transport users than those who had not used pt in the last 12 months
How would you access the new system? | Potential users were twice as likely to use the phone in contrast to assess by the internet
For what types of journey would you use the service? | 95% for leisure, 30% for business, and 20% for commuting;
 | 85% for long distance, 50% for local,
 | 80% for multi-mode journeys,
 | 90% for the train, 70% for the bus, 30% for coach, and 16% for underground/tram/metro
How would you prefer to get information at the end of a long distance journey | 30% put on hold while it is retrieved on their behalf, 60% preferred to be put through to someone with local knowledge
What sort of hours would you expect to use the service? | 06:00 to 08:00 (35%); 08:00 to 20:00 (85%);
 | 20:00 to 22:00 (55%); and 22:00 to 06:00 (25%)

Table 1: Harris survey of public transport information

In summary, it may be indicated from the above research that the public's access to information could be improved. If it were improved, most would use it. However, coupling these results with the previous section suggest that little evidence exists for direct revenue generation from the increased or improved provision of information. However, until such a system exists, the interpretation is somewhat speculative. The research in this thesis details how such a system might be enabled.

2.2.2 Traveller Requirements

The above sections have demonstrated a number of issues affecting passenger's access to and interpretation of information. But before a passenger can have access to the information, it has to exist in the first place. The information must be comprehensive as individual travellers have different requirements depending on their journey profile, where their profile depends on the journey length, purpose, and so forth. For instance, the journey could be quite long or just a short trip to the centre of
town. They could be familiar with the local area or come from afar and be complete strangers. The
information may be time sensitive if travelling to a meeting.

In an ideal situation, the traveller might wish to access a portal where he or she enters into a short
dialogue with another person or interacts with a device. In either case, the 'device' then searches for and
retrieves all relevant information to enable a journey decision to be made for travel between one point and
another, i.e. a 'one-stop shop' (Prescott, 1999).

Given the variability in needs and locations, a number of portals or 'one-stop shops' are necessary, and
"locally" accessible by whatever means is appropriate for that traveller at that time and in that place
(Wilkinson, p43-52, 1998). For instance, the traveller may have access to a phone, television, or
increasingly, the Internet. Alternatively, the traveller may be limited to requesting the information
verbally, looking at a timetable at the bus stop, in leaflets, or in pamphlets. Also, they may listen to the
radio or visit a kiosk.

Wilkinson et al (1998) go on to state that a number of technical reasons exist for this mismatch of needs
with what is provided apart from many other issues discussed previously and later. As stated in the
objectives, no system was found which exchanged information between systems. Also, no system has
access to all modes, nor is any system independent of computer architecture or transport information
supplier. However, the content of the information contains the same fundamental elements: i.e. catch the
22 bus at the corner of the High Street and Commercial Road at 10:00 on a specific day; go to ..., then
change to the ... and get off at the intersection of Northern End Road and Sycamore Close, also indicated
by a Red Lion Public House. Thus, what is necessary is a "mass customisation" of information regarding
public transport services, which matches the traveller's profile. As stated in the objectives, what is
lacking is a collection of procedures to enable this expectation to become a practical reality, using
existing resources in so far as possible.

### 2.2.3 System Requirements to Define and Plan a User's Journey

As can be confirmed from a small survey of electronic or internet-based journey planners (see Appendix
B), any journey planner needs the traveller or potential traveller to specify a small list of elements or
parameters so as to define a specific journey. The following table summarises these minimum necessary
parameters for a journey definition from the system's perspective.

<table>
<thead>
<tr>
<th>Input Criteria</th>
<th>Number of TripPlanners</th>
</tr>
</thead>
<tbody>
<tr>
<td>Origin</td>
<td>ALL</td>
</tr>
<tr>
<td>Destination</td>
<td>ALL</td>
</tr>
<tr>
<td>Date</td>
<td>ALL</td>
</tr>
<tr>
<td>Time</td>
<td>ALL</td>
</tr>
<tr>
<td>TimeType: i.e. Depart After or Arrive By</td>
<td>ALL</td>
</tr>
</tbody>
</table>

Table 2: Summary of surveyed planning criteria: Input
A definition of the journey enquiry is thus defined as an origin, a destination, a date, a time, and a type of journey enquiry. For instance, the traveller needs to state whether the given date and time are the time that the traveller wishes to depart after, or the time when they wish to arrive by. Other parameters are sometimes included such as language of display, avoid journeys by car, rail, coach, or that travel through a tunnel. Alternatively, parameters included a limit on the number of options returned. Thus, these parameters typically related to how the traveller desired to see the results.

<table>
<thead>
<tr>
<th>Output Criteria</th>
<th>Number of TripPlanners</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>ALL</td>
</tr>
<tr>
<td>Route</td>
<td>ALL</td>
</tr>
<tr>
<td>Price</td>
<td>1</td>
</tr>
<tr>
<td>Foreign Language Option</td>
<td>6</td>
</tr>
<tr>
<td>Could handle more than one mode</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 3: Summary of surveyed planning criteria: Output

Most of the surveyed journey planners usually returned an itinerary of the journey details to the traveller. These details typically including the departure and arrival points, dates, times, service numbers, and service operators. Sometimes the journey planner included the class of travel depending on mode, as well as the level of service, and whether reservations were needed.

2.2.4 Importance of Response Time

Thus far a number of issues and traveller requirements have been discussed. However, if the information service is interactive, a "short" response time for the service is also a crucial requirement, especially for journey planners (Potgraven, 1996). First it is necessary to define what is meant by response time. On the one hand, there is "user" time, that is, the time taken for the user to understand the information displayed, make a decision, and communicate it to the machine via taking an appropriate action, should he or she wish to do so. On the other hand, there is the time the machine (or collection of machines) and its software take to respond, as programmed, to an action by the user.

Research studying the user time might look at presenting the information so as to minimise the time taken to comprehend and take an action; examples include INFOPOLIS I and II. In the research described here, the focus was on looking at enabling the decision to be taken, i.e. the journey planners' perspectives.

As no known collection of methodologies existed to achieve the aims set out in the objective of this thesis until the subsequent (1999) creation of EuroSpin and EU-Spirit projects, the research in this thesis concentrated on determining these methodologies. If it was successful, it could only then measure overall journey-planner response-time as a basis for future projects and research. It will be shown in Chapter Seven that in the meanwhile, it is only possible to measure the response times of individual processes in or rather, between individual components of journey planners.
However, ambiguity in the discussion of the “system response time” still exists which can be clarified. For instance, a single journey planner covering a specific area—sometimes called a “region” (EU-Spirit, 1999)—can be expected to respond “instantaneously”. As measured, this means under a second (DB AG, 1999). Yet a complex journey can require analysis of a great many possibilities, especially if the transport service network is dense and frequent (i.e. London). Thus an “immediate” response here might take at least seven seconds (Coucher, 1997). Both of these responses are from journey planner systems aimed at call centres or what is sometimes termed the “professional user”. Discussions with Paylor (1998) and his colleagues suggest that 5 seconds is reasonable. Other members suggest the more conservative estimate of 15 seconds (Houghton, 1998). Research studies for Eurospin state a desire to reach a response time below 10 seconds for users (Booth, pp32., 2000). More conservative estimations come from representatives of local and national government at 30 seconds to a minute (ATCO, 1998). The latter estimates are for “Internet” users. However, if the design of the system requires use of the Internet, it is difficult to distinguish between professional and Internet users. One company (Henrig, 1999) suggests that if there is interaction to distract the user, some flexibility in user response time requirements exists as the true processing time is masked.

The main points are that the perceived acceptable time would vary according to enquiry type. For instance, if the request is for a local journey, the system should respond quickly. If the person is waiting on a telephone, the response time of the system needs to be quicker than if accessed by the Internet. Whilst limited success is evident for charging the user premium rates (Lynn, 1996), local journeys are inexpensive (i.e. less than the price of a premium rate call at 50 pence per minute, average call length being two minutes—see Lynn, 1996) and thus unlikely to service premium rate calls. If the journey is a long distance journey, more patience might be tolerated for Internet users, but still very limited for phone users. If the person is using the Internet and waiting for a response, response time can be more forgiving, but research by Nielsen (2000) suggests that even ten seconds is a long time to be staring at the screen unless there is some distraction. Research by KeyLabs (2000) suggests users perceive eight seconds as a long time.

When review of available technology for this thesis was conducted in 1995, the above estimates were unavailable. However, when the response times from individual journey planners were measured, they were found to be “nearly immediate” in most instances (less than a few seconds, except in the case of EDS' ROUTES journey planner, using the PARIS algorithm, for London Transport; Coucher, 1995). These measurements suggested that the sum of responses from several components, combined so as to meet the multiple modes and cross boundary criteria set out in the objective, might be less than a minute or two.

It is important to note that if the “regional” public transport service network is dense and complex (i.e. London), then the database for a “region” can be very large. In these cases, the journey planners tested reacted more slowly than databases representing the smaller networks, which may have been limited to a single mode, had less granularity (i.e. perhaps only timing points, instead of all stops) of data, or had
fewer services in the area. At some point, depending on as yet unknown criteria, it might be more efficient to build some "intelligence" or "knowledge" into the "journey planner" and perhaps call it a "navigator". This intelligence might allow the "navigator" to get the relevant information more quickly than by using standard search techniques. This concept of intelligence might take the form of what is sometimes termed a "broker", as it decides what information is relevant in which database on behalf of the traveller. However, before discussing this further in Chapters Three, Four and Five, it is necessary to review existing services, a number of issues with respect to data, the system architecture, and data transport.

2.3 Existing Services

Where travel information exists at all, the quantity of the information can vary considerably. Some information is quite specialised. It might only show the schedule information for one transport operator, or to a particular destination. A local transport information service might also specialise in one language, or include just one set of features and styles of presentation. The following section considers the services available, thus giving an insight to potential sources for data, examined in the subsequent section.

2.3.1 Historical provision of Public Transport Information

Imposed market mechanisms on the public transport sector in England have led to undesirable effects on the provision of schedule and fare information. Between 1985 and 1995, the government privatised and deregulated both the bus and the rail transport sectors (DETRc, 1998). This resulted in the fragmentation of route, timetable, and fare information, as bus and train companies began to advertise only the schedules of their own services. Publishing of timetable information in areas like Birmingham and Manchester was discouraged altogether by local bus companies, because competing operators could then run a bus minutes before another bus company's own scheduled service (GMPTE, 1996). As services have high frequencies on a few routes, the necessity of a timetable was questioned. However, where one transit operator controlled a significant portion of the local market, publishing competitors' timetables in addition to one's own increased overall transit use and occasionally paid dividends, as in Wales (Gibbard, 1998) (Warman, 1999) and for National Express (Brown, 1998). Publishing in other areas was not possible due to a lack of budget allocation (Slevin, 1996) or not necessary as services are non-existent or very infrequent (postal services) (Kent County Council, 1997), as the case in sparsely populated areas.

From the perspective of the operators, publication of intermediate points is discouraged (ATCO, 1995-2000) and DETR (g 2000). If a transport service operator publishes a time at a particular stop, the service is accountable to be at the stop at that time. This requirement, given a five-minute grace period either side of the published time, is sometimes strictly enforced, regardless of mitigating factors such as traffic congestion. Thus, this measure of accountability sometimes works against the traveller, as the transport operator may take the view that only the beginning and end of the route will have specific times, so as to
a avoid penalties. Sometimes extra timing points are included between the start and the end of a route, even though the bus does not stop at these points. (DETR, 2000)

Another relevant example occurs in rural areas. Consider a bus route that is 14 kilometres long with timing-points every kilometre or so. If there is a bus stop at shorter intervals over part of the route, say at “normal” intervals of 600 feet or 175 metres (White, 1996), this could imply that as many as six \((1000/175)\) bus stop poles are without a reasonable approximation of departure time. Given the choice of walking the required distance to the destination, taking say 30 minutes, or waiting up to an hour and then paying for a bus journey to take ten minutes, perhaps longer as it is caught in traffic, able bodied persons may elect to walk. Were scheduled information to be available at the bus stop or via other means (a mobile phone, word of mouth, et cetera), the potential traveller may be able to improve the utility of their time. Thus, from a traveller’s perspective, these practices of penalising early or late arrivals may prove unhelpful. (DETR, 2000)

Again from the bus operator’s perspective, another reason exists to discourage publication of times. Competing operators are able to run a service a few minutes before their scheduled service and thus the original transport operator potentially looses many patrons, as may be the case towards the end of two competing routes to a city centre. Whilst this has the effect of grouping services for the traveller at over a short band of time, and potentially offering a choice of service operators and fares, it may also emphasise the perception to the traveller that nothing comes for ‘ages’, and then all the services come at once (DETR, 1999, 2000). Observation suggests that this practice encourages some potential, perhaps more affluent users to find other means that are more predictable, whether or not they had the scheduled time at the onset.

Where budgets allowed and segmentation of the local market necessitated, some local government councils and authorities attempted to provide consistent, reliable and comprehensive route, frequency, and sometimes timetable information, both for socially necessary public transport services that the council supported, and for profitable services in the county. No local authorities provided fare information for buses, although the national rail and coach companies provided fare information for trains and coaches respectively. (DETR, 1996-1999)

For the above reasons, and as providers of transport services themselves, councils began to get involved in publication of bus information. In some cases, this involvement had the effect of removing bias and improving the consistency and quality of information available, and perhaps enabled a traveller to have knowledge of services from more than one company in the same booklet. This knowledge removed the possibility that only the services of one operator were published, yet two services passed by the stop.
2.3.2 Traditional Publication Methods

The methods typically available, i.e. the printed timetables, are the methods that would be used most frequently as shown in Table 4 (Railtrack, 1998). Interviews with local operators suggest that many people use 'word of mouth', and then resort to other sources (Curtis and Knockles, 1996).

<table>
<thead>
<tr>
<th>Sources that would be used to find out information on services</th>
<th>Train % Services</th>
<th>Bus % Services</th>
<th>Plane % Services</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local bus routes and timetable pamphlets</td>
<td>57</td>
<td>65</td>
<td>36</td>
</tr>
<tr>
<td>Teletext</td>
<td>46</td>
<td>50</td>
<td>45</td>
</tr>
<tr>
<td>National Telephone Inquiry Service</td>
<td>33</td>
<td>31</td>
<td>30</td>
</tr>
<tr>
<td>Public Electronic kiosks at bus/train stations and airports</td>
<td>22</td>
<td>24</td>
<td>14</td>
</tr>
<tr>
<td>Public Electronic kiosks at shopping centres and libraries</td>
<td>20</td>
<td>25</td>
<td>13</td>
</tr>
<tr>
<td>Internet (WWW)</td>
<td>14</td>
<td>14</td>
<td>13</td>
</tr>
<tr>
<td>Public Electronic kiosks at your place of work</td>
<td>8</td>
<td>8</td>
<td>6</td>
</tr>
</tbody>
</table>

Base: Employees at sites of 50+ people (307) (Source: Railtrack, 1998)

Table 4: Sources of public transport information

County Councils and industry often used traditional methods of timetable production before 1995 based on lithography. With the widespread adoption of computer technology, and more than an order of magnitude increase in storage capacity and data processing speed with respect to an equally large decrease in price, opportunities arose for innovation. (Slevin, 1995-1998)

A direct result of this computerisation is that councils are moving away from printing methods using hand based pictures and typesetting as above, and producing printed books and pamphlets and other materials using computer technology. They are commissioning digital images, which can be easily and inexpensively stored and retrieved. They are also producing electronic timetables from digital databases. (Slevin, 1995-1998)

The impetus for these changes comes from an inherent and major disadvantage of traditional publishing systems, i.e. information is often incorrect and out-of-date before it comes off the press. This problem results from the lag in time to complete the mechanics of publishing timetable booklets. By law, a transport operator may change the timing schedule of any part of their service as long as they give the council 42 and 56 days notice, depending on area. Printing of the timetable booklets follows this period, and may take several months. This "delay" is partly due to writing the specification, tendering, and editing. With each edit, typesets have to be physically changed. This can take considerable amounts of time. Once printed, distribution is delayed by payment cycles and transport of the timetable booklets to the place of sale. Whether given away, sent to subscribers, or sold as magazines, there is yet another delay in the process. Thus, these processes slow down printed production of pamphlets and booklets: registering of the service; changes to the service; printing of the booklet; and distribution of the booklet. (Slevin, 1995-1998), (DETRA, 1996, c, 1999, g, 2000, h, 2000)
A second issue is cost of the timetable booklet. Most of the cost is borne by the supplier or council and this may cost several thousand pounds (Slevin, 1997), (Capes, 1991). Budgets for some counties for this exceeded one hundred thousand pounds. In trying to recover some of this cost, the council may allow a limited amount of advertising within the booklet. The cost of timetables to the traveller may be free or cost 20 to 50 pence for one booklet. For this investment, the timetable may be valid for a particular transport operator, or for a wider area as provided by the council. Further, it will only be valid for a particular period, which for rail had typically been every six months, changing in late May and early September. Because bus companies change their schedules ten times a year or more (DETRc, 1999), it is difficult to state accurately what time period the service is valid for in printed publications, especially with different validity periods for services crossing boundaries or school districts.

A third issue affects the usage of timetable booklets. The simple act of obtaining a timetable may be difficult (Railtrack, 1998). A traveller needs to first determine what timetable is necessary. Thus, the traveller needs have an idea of where they wish to go, when, and where they wish to depart from. It is useful to have some idea if a transport operator provides a service between these two points. Next, the traveller needs to determine who provides timetable booklets. Typically, either the transport operator or the council provides some sort of booklet (GMPTE, 1998). Once the traveller has the timetable, other actions are necessary to determine how to use the particular transport service. These are reading the table; finding the relevant service; and checking the details. Whilst the commuting traveller may know the schedule by heart, these tasks can be much more difficult for others. Deciphering timetables can be time consuming. (DETRa 1996).

2.3.3 Innovations in Information Provision

Whilst the traditional methods of distributing timetable information continue, existing and innovative technologies supplying public transport information are further influencing trends towards electronic and database publishing techniques. Madame Bevulle of RATP (1995) reports several innovations. Railtrack further found a potential for electronic technologies to penetrate the traditionally paper-based public transport information market (Railtrack, 1999) (MacFadden, 1999). As indicated in Table 4, these innovations include Teletext, telephone enquiry services based on databases of schedule information instead of text and pictures, kiosk services also based on personal computer technology, and interactive enquiry services using Internet technologies.

Early research in the TURTLE project (1994) promoted the advantages of Teletext as the way forward (Teletext is simply a display of a page of text and is limited to preformatted pages of text). An advantage of Teletext is its wide distribution, and once the TV licence has been paid, it requires no additional subscription charges. However, in 1995, the actual television unit necessary to receive Teletext cost more than a normal television. With a television, an annual subscription is necessary and this was about £100 per year (TV Licensing, 1997). Thus, whilst Teletext would be useful for the static display of schedules, it would not be possible to support an interactive exchange of data. Therefore, the TURTLE (1995)
project later concluded that Teletext might not be as useful as first envisaged due to developments made possible by the arrival of Internet technologies.

Numerous councils have begun Telephone Enquiry Services, either in an attempt to replace or supplement printed services. An advantage of Telephone Enquiry Services is that they have a wide and inexpensive area of coverage, with the price of a local call being approximately five pence inc. VAT per minute for the traveller (BT, 1998). A second advantage of this service is the ability to support a fax-back service. However, the supplier of a telephone enquiry service must provide trained customer service representatives, a dedicated "local call" line rental, and information systems support for this service. Thus whilst inexpensive for the traveller, the majority of the expense is borne by a local authority, county council or others.

To allow greater access to information at the location of the services and at public places, councils have also begun to experiment with Kiosk services. Kiosk services are sometimes provided by councils and operators as an inexpensive means of providing advice to the traveller. These units can cost around £5,000 per year and require frequent maintenance. (Hants County Council, 1998) Further, vandalism is an issue which increases the need for a more robust and expensive unit (Moore, 1999).

Available methods used by Buckinghamshire (Slevin, 1995) and Hampshire county councils (SCC, 1994) had geographic area limitations, in that it was not possible to plan a journey to a detailed destination outside the defined area. For instance, a local trip took the traveller to the local rail or coach station/stop, and from that point a rail or coach station in the community of the destination. However, once at the rail/coach station the traveller was left to find his or her own way to the end point of the journey. Furthermore, it was possible, even probable, perhaps, in areas with densely located rail or coach stations, that the rail or coach station associated with the destination was not the nearest station to the point of final destination. Four research reports by McCormack (1994) looked at the application of "object oriented" technology to enable a single database (i.e. single area) journey planner (using the UNIX operating system) for public transport. It documents this issue with not-the-nearest-rail station too.

It seemed unlikely that one council would pay to support another council's systems to allow travellers a comprehensive end-to-end journey. Additionally, each council had different suppliers with different characteristics or features. These differences in characteristics included software and presentation aspects, and thus one supplier's system could not talk to any other heterogeneous system.

London Transport (now Transport for London) also commissioned a journey planner from EDS and then to ICL (Coucher, 1995/7). Their system was based on one supplier, and attempted to integrate all modes with London (walk, bus, tube, rail, and coach), not only in text form, but with maps also. The approach taken was to create a large database, and run everything from a mainframe. London Transport have since shown interest in the research described in this thesis as they seek to resolve issues of a "slow" response
(several seconds), capacity limitations (enquiries per second, and cost per enquiry), and integration with the rest of England. (Coucher, 1995) (Bourne and Harris, 1998-2000).

2.3.4 Combining innovations

As printing technology, telephone enquiry, and kiosk systems began to evolve, the need for a common database to support these systems became evident. Concurrently, Internet technologies became increasing available. Some councils and several political bodies in the United States began to exploit these technologies and couple the advantages of relating these innovations.

The primary advantage offered by storing timetables in a database was that the information could be extracted to match several purposes quickly, whilst being more up-to-date and perhaps more accurate. For instance, a timetable booklet for a scheduled bus service could easily be printed and in smaller batches. The same database could be queried for a single bus trip on a particular day and time to answer queries by travellers calling through on telephone enquiry services, or using a kiosk. Furthermore, by using such a database, the whole timetable could be “published” using Internet technology and accessed by anyone with a browser anywhere in the world around the clock at minimal cost to the supplier; the requesting traveller also pays the cost associated with any printing. (Slevin, 1997)

With adoption of these innovations, councils and operators began to experiment further. With the possibility of requesting a bus trip between two points on a timetable schedule, it should be possible to have computer software formulate methods to query databases so as to combine two or more bus trips for a more complete bus journey. Further experimentation (see section 6.1.2) proved this was possible for a small database (such as a county, national rail or coach) and could be done sufficiently quickly to support telephone enquiry systems, kiosks and Internet technologies. (Slevin, Houghton et al, 1996-9)

As these systems developed, and access to the Internet and Internet-supported-services increased, industry and councils supposed that the availability of these services to travellers would also increase. In so far as trends allowed, national transport services such as rail, coach, ferries, and airlines began to invest in these technologies. Computerisation of schools and offices increasingly gave larger portions of the population access to Internet technologies, increasing the number of potential travellers with access (Lyons, 1999). However, expanding these developed systems beyond the scope of a relatively small geographical area of a county or similar size database proved to be quite a challenge.

2.3.5 Application of innovations

In an effort to marry database and Internet technologies, several types of pre-trip, public transport information systems came to market. Travel agents refer to on-line availability and reservation systems such as Galileo, Sabre, or WorldSpan. Other examples of on-line, fee-based services include “Tel-Me” and previous travel products by CompuServe. In buildings and rail stations, examples of stand-alone or
kiosk systems in England are sometimes found, including WS Atkins's TRIPlanner, Pindar Routel's Travel Guide, and London Transport's tube planner. Foreign examples of kiosk services include the GeoFox journey planner in Hamburg, Germany and SNCF systems in Paris and in major rail stations throughout France. Examples of single license, single user journey planners using personal computer software are on sale from British Rail, S.N.C.F., D.B., the Automotive Association, Ordnance Survey, Microsoft and others.

ROMANSE (Mansfield, 1995) looked at the application of this “object oriented” technology towards enabling a multiple database, multi-modal journey planner, but used a Windows® operating system. Unfortunately for the traveller, each database was accessed separately, as was each mode. For instance, the traveller could travel by bus, coach and rail, or by car. Integration between bus and (rail and coach) was via fixed exchange nodes. However, in practice, these exchange points consistently lacked much detail, especially for the 'new' user. Physical trials confirmed this (Southampton City Council, 1994). This was also true of an Internet based system by Buckinghamshire (Slevin, 1995). Consideration for air or ferry journeys was limited.

Many journey planners were modal in nature. In the early stages of the literature review, only the ErOpIt Windows 3.11 based journey planner for the Netherlands could combine modes (buses and trains for instance). This has changed somewhat with the introduction of HAFAS from Hacon (www.hacon.de), and comparable products by the makers of Elektronischen Fahrplan Auskunft (EFA). A Deutsche Bahn railway journey planner (now based on HAFAS) provides both time and price data for domestic services, and times for many international railway services (Briggs, 1996-9).

The GeoFox stand-alone/kiosk journey planner in Hannover, Germany incorporates many significant features. For instance, it displays time and price together for local journeys (personal trial, 1996). For certain local routes, it shows a journey combining bus and rail modes. Finally, it allows the traveller certain preferences, such as whether to exclude travel in tunnels or via water. However, its application is limited to the area of Hannover.

Some of the above examples have a fee attached with their usage. When free, the information may be biased if provided by the supplier (i.e. Railtrack, National Express, and First Group). Only services sponsored by local government included an element of impartiality. However, where a traveller can get information on more than one mode, say a journey that includes trips on buses and trains for instance, information was limited to the geographic area of the authority sponsoring the service.

2.3.6 Public Transport Resources on the Internet

As a result of the above environment, several county councils, including Buckinghamshire and Hampshire, in association with independent data service supplier companies, began to invest in various services for travellers to maximise dissemination of timetables at a minimum cost. These services
included digital or "on-line" timetables of bus stop information for particular routes, use of computerised journey planners allowing a traveller to plan trips, via public transport only, within a county, and telephone enquiry services. A comprehensive database of timetable schedules made these services possible, and evolved by conversion of paper-based publishing to digital publishing. This conversion partially resolved obstacles created by frequently changing bus services. It also enabled publishing on the World Wide Web (WWW) and other Internet mediums.

Without compatible systems, councils tried to extend county-based systems to provide a wider (national) service as an interim measure. Longer journeys (e.g. at least two trips and an interchange) may cross county boundaries as some bus and train services operate both within and to, or through, adjacent or adjoining counties. In building journeys from services, county councils’ efforts to hold timetable details increasingly duplicated service data from nearby counties’ databases. Further duplication arose as each council openly encouraged data service suppliers to include national rail, coach and air timetable data using digital technologies. This duplication was costly and thus the impetus to overcome geographic, mode and supplier limitations increased.

Besides duplication within individual county councils’ systems, the national rail infrastructure company (Railtrack), its constituent train operators (ATOC), and national coach company (National Express) all provide journey planning services via the World Wide Web for their services only. Yet with the number of bus services operated by Train Operating Companies (i.e. Romsey to Winchester), and number of train services operated by coach or bus operators (i.e. Gatwick Express), the need for journey planners which could plan a journey via more than one mode of transport became increasingly self evident. Coupled with the increasing number of county council or unitary authorities either planning to develop or already operating a journey planning service available via the Internet for their local area, this duplication of data was consuming significant yet already scarce resources of the councils (Slevin, 1995-8). Examples of counties with journey planners include Bedfordshire, Buckinghamshire, Cumbria, Greater Manchester, Hampshire, Hertfordshire, Kent, Lancashire and Surrey. (Marshall, 1997). The Welsh produced a similar system for their call centres in August of 2000. The British Airports Associations has an in-house system for its employees at Heathrow which covers a 50 mile radius, and includes a majority of Greater London (Houghton, 1998).

Elsewhere, the USDOT (Casey, 1998) suggested that WWW based pre-trip information covers "a wide range of categories". These categories include "transit routes, maps, schedules, fares, park and ride locations, short term passes, points of interest, weather" and so forth. The USDOT further stated, "often, this information can support itinerary planning, which can provide information on an entire trip from origin to destination, even if it involves multiple modes" (Casey, 1998). Bernard (1995) suggested more simply that four basic methods may influence the travellers' decisions via publishing public transport traveller information. Samples of these methods can be found in Appendix A. These methods include:

- supporting information such as routes, maps, phone numbers, and so forth
• published schedule information
• published fare information
• published journey planning information

Route information describes the route of a particular transport service. For instance on the route illustrated in Appendix A, route 4 of the bus service goes from the General Hospital to the University and then to Portsword (Southampton, England).

Map information (see Appendix A) allows the user to see a graphical image, which depicts an area, and may allow the user to navigate these maps by linking images together. (MultiMap.com, 1998-9)

Weather information (see Appendix A) takes the form of text descriptions of local forecasts for a particular area or region. These can be useful in aiding the traveller to know how to dress or otherwise prepare for the journey. (The Met. Office, 1998-9)

Service level and disability information (see Appendix A) also takes the form of text descriptions of what services are available at particular points along the journey, such as changing facilities and whether the platforms can be accessed by ramps or lifts. (MUST, 1996-8)

Schedule information (see Appendix A) takes the form of text and is typically preformatted and appears just as a timetable would on a printed pamphlet. An example of journey planning information is also in Appendix A (Buckinghamshire County Council, 1995-8).

Fare information (see Appendix A) is sometimes included with schedule information, such as at the CalTrain site (http://www.caltrain.com/). However, determining fare information for display with respect to itinerary planning is not yet practical due to inherent complexities (Mason, 1996).

Limited attempts at 'real-time' (i.e. bus 31A arrives in 10 minutes) information have taken place with research and deployment being carried out by Hampshire County Council, London Transport, and others. Whilst pioneering in nature, they were sometimes unfortunate enough to tell the traveller that the bus was just to arrive when it had just left. Advice on how a traveller might cope with unplanned delays or events occurring in the hour or so before departure ("contingency information") is still seen as a future area of research.

2.3.7 Summary of Available Information Services

Prior to 2001, operators of public transport services in England and Wales were not required to provide basic information such as routes and times about their services. Even when information was provided to the council, the routes and times of a transport service was described using different methods. Some

2 http://www.tag.co.uk/must/
operators used a listing of ambiguously named stops, starting at a given time (the first departure time that day) and having a set frequency (i.e. once an hour). If the service is frequent enough not to warrant a detailed schedule, transport operators have sometimes taken the opportunity to simply publish the frequency and the route, either in a graphical or text form; sometimes both are printed. For instance they might just draw a line on a map and state that the service was once an hour. If the service is infrequent, no details may be available. Some services are hail and ride. In some urban areas (i.e London, Southampton), it has been observed that scheduled services sometimes mimic a "hail and ride" should a potential traveller hail a bus "near" a stop, though no known documented evidence for this practise exists.

In contrast, a traveller may seek to optimise his or her time of travel. As a result, they may increasingly need to plan the journey more carefully. This need increases when services are infrequent, i.e. in the early mornings, from early evenings till the last service, and at weekends when "normal" frequencies of twice an hour to every few minutes are reduced to once per hour or two. Should the traveller's journey require a change of vehicle or mode, planning often requires precise times for a particular stop. If the stop is not a "timing point", as defined by the Traffic Commissioner, the traveller must use interpolation to estimate the true time if the transport operator, or the council on their behalf, does not do this for them. As publication is expensive, many transport operators did not consider it in their interest to interpolate times for the traveller (ATCO, 1995-9).

Thus, it was seen as a step forward in England to investigate the feasibility of one data-supplier-company exchanging data with another. Even if England had the same environment as the Netherlands, it is likely that this functionality would still be necessary for 'international journeys. The Netherlands has a distributed data storage environment. As such, it was not necessary for their consortium to develop a generic protocol allowing transmission of data between competing organisations. Any necessary data transmission occurred within the organisation; thus an intranet protocol was sufficient. However, with the occurrence of data transmission via networks outside one's own wide area network (WAN), it may be useful to develop a system of data transmission utilising the more widely available Internet protocol.

It was clear that many parts of the United Kingdom were ready to share common information resources covering rail, coach and air, and each other. However, an extensible system should be to cope with the fact that numerous counties do not yet have databases and journey planners. Alternatively, if the network cables were cut and a region was effectively removed temporarily, the system would need to have the ability to provide contingency services. Similarly, co-ordination of a "national system" so that it all went live at the same time could be disastrous politically should anything go wrong. Ideally any system would potentially need to allow for incremental implementation. In an ideal situation, it would also enable system suppliers to collectively spread the demand (the Internet load) for information requests over the available resources, and minimise the need to duplicate data from another area.

To conclude, the available methods of Public Transport Information dissemination are as follows (Marshall, 1998-99):
- Public Access Terminals or kiosks (Thru-the-Glass or Touch Screen)
- Telephone Enquiry Systems, and with fax-back
- Passive Display Systems (Monitor or LED) at station stops
- Timetable Printing
- Desktop PC-based Enquiry Systems
- Enquiry Systems using the Internet

An important point is that the last method, Internet based access, can support all of the above methods.

2.4 Existing Sources of Journey Planning Data

Whilst some areas might see an increase in public transport usage, and others a decrease in public transport usage, overall it is hoped that public transport usage might improve by enhancing the quality of public transport information, especially between geographic areas with administrative or database boundaries. Whilst the requirements of a journey planner with respect to a user have been stated above, the system of computers needs to have data available for it to develop and offer choices to the user. It also needs data to compare against the user's choices. Thus the following sections examine what data exists, its accuracy, the location of the data with respect to time, its availability, and the presentation of the data to similar computer systems. Answering these question requires a detailed investigation of the locations of the data, a global architecture for the data network, the physical transport of the data, and to a limited extent, the usage of data which might be necessary to support these decisions. All investigations except for the first are addressed in later chapters.

2.4.1 Data Requirements for Journey Planning

Returning to the data necessary to service "traveller requirements", examination of the journey planners surveyed (Appendix B) suggests that the internal data requirements for any one journey planner might include three types of data resources to assist the traveller.

Alternatively, it is perhaps easier to look at what a typical journey planner returns as a solution. In its most basic form, the response is "start from ... at this time and date, go to this point and wait for service A, get on that service and ride it for X minutes, getting of at ..., and then walk to this (exchange) point and wait for service B to arrive at time, and then ride that service for Y minutes, then alight and walk to the final destination. Looking at this in detail (see Figure 1) suggests that a computer needs a collection of "services", "exchange points", and "normal points".

Services in this research are composed of trips. These trips can be pasted together at exchange points to form a journey. The trips and journeys start and end at places, which are logically related, in some unknown manner, to the melange of political hierarchies discussed earlier.
Places are also geographically related in "space" by a set of grid co-ordinates. If these places had a global system of characteristics such as latitude, longitude, and an altitude, then it would be possible to relate them to one another.

![Diagram of Journey using two Services](image)

**Figure 1 : Detail of a Journey using two Services**

Thus one type of data resource would be a list of stops or places. This origin or destination list is sometimes called a 'gazetteer'. The gazetteer has two functions. In this research, as in others, EU-SPRIT (1999), the gazetteer provides a list of 'places'.

If all the data is not held in one data storage area, and in a universal system, it is a foregone conclusion that it would not, it seemed intuitive for a need to list available collections of place or location data. Further, these deposits of location information data need to be referenced to each other in some manner.

To relate places to services, it followed that another function of a gazetteer is to associate these lists of names of places with other data resources, which have further information of relevance to the place. So a list of references to further information is necessary also. This ideology was unique to this research until EU-SPRIT in 1998 (published in November of 1999). This point is revisited in Chapter Seven, as its method of implementation is expected to directly affect a largely unmeasured effect on extensibility.

Research by EuroSpin (1999) seems to use grid co-ordinates as a means of determining further information, and this will not be entirely consistent with methodologies discussed later in this research. As before, the EU-SPRIT (Schnittger, 2000) method in this function is similar. It however requires that this "gazetteering" feature, which relates a reference or index to further refinements of location data, be included within their "object request broker" instead of being more openly available as this thesis shows in later Chapters.

A second type of information arises from the need to transfer between modes and services. If the traveller is to change from one mode to another, e.g. defined as bus to walk and so forth, a listing of exchange points is necessary. For the sake of consistency with other publications, this listing was named a 'registry'. This name relates to the two functions of the registry. One function is to provide a list of interchanges or exchange points where a traveller transfers from one mode to another. This can either be
simply any point where a traveller boards a bus, coach, train, boat, aeroplane, or another vehicle, or it can take on the meaning of transferring from bus to a coach for instance. The second function is to associate these interchanges with the data source that can supply further information. This data source could take on the form of a reference as in the gazetteer above.

Perhaps the key data resource necessary is that of the services or actual schedules. This schedule or timetable data consists of the actual times and points, and a network of linkages which make up the routes. From this data, a journey can be determined and displayed to the user.

2.4.2 Names of Places Data

The following table summarises most of the known data that existed at the time of writing. Not all sources of information are included, but only the most relevant. They relate to three different categories of available data: those that relate to places, those that relate to exchanges, and those that relate to schedules and fares.

Several organisations hold lists of villages in the United Kingdom. The national organisation responsible for geographic naming has about 45,000 village names in its archives. A national road organisation has 31,000 village names. Other data archives also hold a similarly approximate number of villages plus or minus a few thousand. Additionally, each county holds their lists of villages.

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Number</th>
<th>Unit</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Communities: i.e. villages, hamlets, districts, precincts, small towns, big towns, cities, and so forth.</td>
<td>35000</td>
<td>Britain</td>
<td>OS – Meridian</td>
</tr>
<tr>
<td></td>
<td>29000</td>
<td></td>
<td>Automobile Association</td>
</tr>
<tr>
<td></td>
<td>57000</td>
<td></td>
<td>ONS – Eng &amp; Wales</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td></td>
<td>ONS – Scotland</td>
</tr>
<tr>
<td></td>
<td>30000</td>
<td></td>
<td>National Street Gazetteer</td>
</tr>
<tr>
<td></td>
<td>30000</td>
<td></td>
<td>OS – Address Point</td>
</tr>
<tr>
<td></td>
<td>22000</td>
<td></td>
<td>OS – NetMap</td>
</tr>
<tr>
<td></td>
<td>42000</td>
<td></td>
<td>OS – National Gazetteer</td>
</tr>
<tr>
<td></td>
<td>35000</td>
<td></td>
<td>Bartholomew</td>
</tr>
<tr>
<td></td>
<td>25000</td>
<td></td>
<td>Royal Mail</td>
</tr>
<tr>
<td></td>
<td>25000</td>
<td></td>
<td>Whereonearth.com</td>
</tr>
<tr>
<td></td>
<td>Unknown</td>
<td></td>
<td>AND (Dutch)</td>
</tr>
<tr>
<td>Points of Interest</td>
<td>5000</td>
<td>Britain</td>
<td>Bartholomew</td>
</tr>
<tr>
<td>Counties or administrative areas</td>
<td>100</td>
<td>Britain</td>
<td>Ordnance Survey</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td></td>
<td>ONS</td>
</tr>
<tr>
<td>places (i.e. bus stops, stations or or buildings)</td>
<td>2000 to 17000 (London)</td>
<td>Per county or administrative area</td>
<td>Pindar Routel, London Transport, WS Atkins, Software Logic</td>
</tr>
<tr>
<td>Transport service operators</td>
<td>10 to 100</td>
<td>Per county or administrative area</td>
<td>Council's records</td>
</tr>
</tbody>
</table>

(Source: internal minutes of Jan 1998 JourneyWeb meeting; and interviews with London Transport)

Table 5 : Statistics relating to PLACES

It may be seen from Table 5 that several different suppliers of data existed for a so named 'national gazetteer'. Of the existing products, OS and AA where the most expensive, and the National Street
Gazetteer was largely incomplete at the time of writing. Bartholomew, Royal Mail, and whereonearth.com were also relatively expensive. The remaining choice of the two Office of National Statistics (ONS) products have largely been combined and therefore show both the most 'locality' or 'community' names. It was also the most inexpensive product by a factor of ten. The ONS data also contained relationships that were advantageous to other data such as grid references and cross relationships. It also has a bi-annual update.

Bartholomew was the only known data source to include specific 'Points of Interest'. The Office of National Statistics and Ordnance Survey both provided free lists of administrative areas (county councils and local authorities). No known comprehensive list of stops is available, nor a comprehensive list of 'points of interest'. However, the Ordnance Survey is heading one such initiative by combining its Address Point product and its road name data. Thus to obtain lists of stops, this aspect of the research proved quite a challenge, and is discussed in detail in Chapters Five and Six.

2.4.3 Exchange Point Data

Exchange point data is largely absent or incomplete except for local purposes (Houghton, Slevin et al, 1998-0), and only came to light when suppliers tried to connect existing databases (see section 6.1.2). Discussions with the members of the industry suggest that the data is available (Table 6) but contains no relationships except for local use. If the local exchanges that are typically within a database were sufficient for mid-distance and national journeys, this would not be an issue. However, the 'national' exchange points can vary significantly depending on date of travel, time of travel and direction of travel, and thus warrant a separate table. For instance, those living in Harrow (London) might use Paddington for trips to Wales and Devon, but Euston or Kings Cross to go North. Residents of Kingsclere might travel to Basingstoke to head South or East, but might expect to change at Newbury if heading West to Wales.

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Number</th>
<th>Unit</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exchange or interchange points (buildings or locations where one changes mode)</td>
<td>5000</td>
<td>Britain</td>
<td>Railtrack = ~2700, National express = 1350, Airports and ferries</td>
</tr>
<tr>
<td>bays or platforms</td>
<td>8</td>
<td>100</td>
<td>Most small rail stations Heathrow, per terminal</td>
</tr>
<tr>
<td>local exchange points</td>
<td>5</td>
<td>per local place</td>
<td>Mostly walk links</td>
</tr>
<tr>
<td>Adjacent county exchange points</td>
<td>5</td>
<td>per local place</td>
<td>Mean number of minor rail, express bus, and coach stops</td>
</tr>
<tr>
<td>National/international exchange points</td>
<td>5</td>
<td>per local place</td>
<td>Mean number of major rail and coach stops</td>
</tr>
</tbody>
</table>

Table 6: Statistics relating to EXCHANGES

It is relevant to state that current databases typically do not actually have true bays or platforms within them. It is presumed that if the traveller needs to change from the coach at Victoria (London) to a train or
bus that they will manage (within five or ten minutes), and as such that level of detail is often missing, or incorrect (Funnell, 1996-9).

2.4.4 Scheduled Service or Trip Data

The number of service operators is a factor, as they provide the services and routes. This information is imbedded in the database of schedule data, but not otherwise available except from paper based maps and bus registrations forms. The latter is being updated to digital but will not be available in time to be included for publication (DETRg, 2000).

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Number</th>
<th>Unit</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Services</td>
<td>10 to 100</td>
<td>per day</td>
<td>Interviews</td>
</tr>
<tr>
<td>Routes</td>
<td>1 to 30</td>
<td>per stop</td>
<td>Interviews</td>
</tr>
<tr>
<td>Fare stages</td>
<td>Up to 8</td>
<td>per route</td>
<td>Interviews</td>
</tr>
</tbody>
</table>
| Number of different adult fares | More than 8    | per transport service supplier | (single, return, next day, peak hour, special, combo, employee, blind)

Table 7: Design parameters for a National Public Transport Information System

Some approximate estimates of statistics for bus services are shown in Table 7. Train and coach services tend to have greater symmetry, but sometimes only marginally. As such, the size of the database table that relates stops to services for train and coach data is sometimes smaller, and may produce quicker search times from a query than a bus database for a county with a comparable number of stops (Ness, Houghton et al, 1995-1999). Thus whilst the train and coach data might have national coverage, it is difficult to imagine a possibility where a bus database could provide both a national coverage and similar response times. Similarly, as the number of stops would be 100 times as large, the number of services to be described could be two or more magnitudes greater in quantity.

2.5 Methods of Data Transfer

Having determined that journey planers have similar inputs and outputs, and having reviewed other global characteristics, this thesis examined of the data requirements within a journey planner. It is necessary to discuss the definitions of the services the data aim to describe. The review below includes consideration of elements found in known 'data dictionaries'.

2.5.1 History of Data Dictionaries

Several data dictionaries were examined. TRANSMODEL (CETE, 1995) was the first dictionary to be examined. Another dictionary was DATEX, published by the DETR (1998). An American dictionary includes TCIP (ITE, 1999-2000), which was not examined in great detail due to its relatively new appearance in publication. EDIFACT, a United Nations standard, existed on the WWW by 1998. Similarly, the data dictionary for EUROSPIN (Tongeren, 1999) was only available as this work was being
published. A proprietary data dictionary includes RJIS (Barker, 1999), which is used within the UK rail industry. Publications for EU-SPIRIT’s have only begun to become available since November of 1999, and thus are not included extensively.

The documentation for Travel Technology Initiative’s Unicorn protocol gives a brief history, and states that “UN/EDIFACT (Electronic Data Interchange for Administration, Commerce and Transport) is a merger of two previous standards - Guidelines for Trade Data Interchange (GTDI) and the American National Standards Institute (ANSI) X.12. Formed in 1986, UN/EDIFACT represents a generic set of international EDI standards, whose rules are published by the UN/ECE in the UNTDID. Included in UNTDID are the syntax rules, conforming to ISO 9735, message and implementation guidelines and code directories.” (TTI, Feb 1998)

This Unicorn documentation further advocates that “UN/EDIFACT operates through world-wide regional boards, each of which has industry specific message development groups. The Travel, Tourism and Leisure (TT&L) group for Europe, EEG 8, operates under CEN (Comité Européen de Normalisation) and EBES (European Board for EDI Standardisation). The most active groups in TT&L are Europe and Pan America.” (TTI, 1998)

DATEX, according to documentation by the recently formed TRIDENT (May, 2000) consortium, builds on EDIFACT by expanding it for use outside the “leisure” and “tourist” markets of transport operators and into Traffic and Travel Control Centres. However, whilst the provision for public transport information is minimal by comparison with the rest of the document, it does build on work by TransModel.

Under a separate initiative, a DRIVE consortium named Cassiope did some initial work that led from Harpist to TransModel, and ended in Titan. TransModel has been proposed as a British Standard since 1997 according to representatives of the Traffic Commissioner (TransXchange Forum, 2000). Unfortunately, the model is designed for the creation of databases and data files. It does nothing for the transfer of data between interconnecting systems.

A small “experiment” was set up in the North of England based on Transmodel. In this experiment, several counties established a common data format by effectively merging two similar formats. This led to the successful transfer of data between the databases of the North of England by no later than 1998. (Moore and Stoner, 1998-9).

The need to transfer rail data between suppliers within the ATOC led to the creation of RJIS.CIF (Booth, 1999), which is again similar but varies slightly, having to concentrate on rail data instead of bus data. ATCO.CIF is used primarily for bus data (ATCO, 1995-9). Having the possibility to transfer what was significant chunks of data in 1995, standards were reputed to be very lean and thus fast and efficient. However, it might be argued that these benefits arose by imprecise data, as only the barest minimum data was transferred.
Whilst these methods allowed and encouraged data transfer, a gap existed in allowing journey planners to interconnect. Journey planners take a small 'token' of data from a large database of schedule information. The basis of this research was that any journey planner might like to take a small token from any other journey planner, as to accumulate all the information into one large database would be mammoth, even with technology as it is in at the end of this research study.

2.5.2 Analysis

EDIFACT, DATEX, TRANSMODEL, TCIP, RJIS, ATCO-CIF and similar "standards" or "protocols" seem to concentrate on data definition. Some of these "standards" include methods for data transmission also, an aspect which is dealt with in Chapter Four.

This research began with an evaluation of the data dictionary for the TRANSMODEL, a project commissioned by the European Commission. Several discussions ensued with the researchers (Roach, 1996). From these discussions, two points are considered important. The definition of a trip and a journey was largely determined arbitrarily according to one member of the project. (Roach, 1996) Also, TRANSMODEL was largely a bus protocol, and as such lacked any information regarding rail, (air, or ferry) journeys (CETE, 1995). The data dictionary was laden with many terms necessary for transit operations, but of little relevance to the traveller. As such, queries run on relational databases built on it could be inefficient with respect to journey response time. On the other hand, if one filtered the journey information and queried just that, a marked improvement might be made.

DATEX is a data dictionary commissioned by the DETR, which references TRANSMODEL heavily. However, the TRANSMODEL data dictionary is roughly twice as long and encumbered by several translations in comparison with DATEX. The following extract describes the correspondence "between the items of the dictionary referring to Public Transport and the reference data model for Public Transport operation, Transmodel". (DETR, Turner, 1996)

"Only the Transmodel concepts of interest in the context of the dictionary have been included. For instance, all the entities describing the basic topology of the network are not included in the dictionary: the entity POINT is only described by two of its subtypes, while the entity LINK is not present at all".

"Some concepts which are described in Transmodel by several detailed entities have been aggregated in the dictionary for simplicity reasons. For instance, the numerous entities describing the various possibilities to define scheduled run times have been grouped into one main concept".

"The definition of entities has been sometimes adapted as necessary, to be understandable in the context of the dictionary. Attributes have been defined explicitly, which is not the case in the Transmodel documentation". (Turner, 1996)
Appendix C includes a table that "provides the relationship between some items of the dictionary referring to Public Transport and Transmodel concepts, included or not in the dictionary." (Turner, 1996)

The TCIP data dictionary is apparently prominent in the United States, but none of the European data dictionaries examined mention it. In examining early drafts of TCIP available via their website (ITE, 2000), no major incompatibilities were found. Documentation regarding the EUROSPIN (Booth, 2000) data dictionary was only just available at the time of publication (Booth, 2000).

Documentation from RJIS, received some time after the publication of the data dictionary for this research, shows a remarkably similar set of key terms, with a few minor differences (Barker, 1999). For instance, instead of just a 'depart by' and 'arrive before' definition, RJIS includes an 'arrive between' and 'depart between' definition also.

2.5.3 Terminology

This research examines and includes a copy of the definitions from the ATCO.CIF file for later comparison with the data definitions of DATEX. It is relevant to remember here that ATCO.CIF, RJIS.CIF and the data dictionary for TRANSMODEL are largely similar, but have differences.

2.5.3.1 ATCO.CIF Definitions

<table>
<thead>
<tr>
<th><strong>A Journey</strong> is the movement of a vehicle (bus, train etc.) described by a chronologically increasing sequence of stopping points and times from an origin terminal point to a destination terminal point. Conventionally this is appears as a column in a printed timetable.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A Service</strong> is a label attached to a group of journeys with (more or less) common stopping points. A <strong>Route</strong> can comprise of one or more services. The <strong>Route Number</strong> is often used to identify the vehicle undertaking the journey to the public.</td>
</tr>
<tr>
<td><strong>A</strong> service is operated by an <strong>Operator</strong>, normally described by company trading name. A route can be operated by more than one operator.</td>
</tr>
<tr>
<td><strong>Stops</strong> define the geographical locations at which events happen during the course of a journey. Often a journey is described by a sub-set of the stops known as <strong>Timing Points</strong> which are the stops defined in a printed timetable.</td>
</tr>
<tr>
<td><strong>Events</strong> during the course of a journey describe what happens to the vehicle at each stop or timing point. Possible events are:- Stops to set down and pick up passengers Arrives to set down passengers Departs having picked up passengers Does not stop for passengers</td>
</tr>
<tr>
<td><strong>Valid days</strong> are days of the week and other special days (e.g. bank holidays, school term time) the journey operates.</td>
</tr>
<tr>
<td><strong>Valid dates</strong> define the first and last date of operation of the journey. In this version of the format full four digit years are used. Previous versions used 2 digit years for compatibility with the BR CIF format. 99999999 may be used to define a journey with unknown last date.</td>
</tr>
<tr>
<td><strong>Clusters</strong> are geographical groupings of stops at which it is possible to change from one journey to another.</td>
</tr>
<tr>
<td>The <strong>Interchange Time</strong> is the minimum time needed to change between journeys at a stop or within a cluster.</td>
</tr>
</tbody>
</table>

Table 8 : List of Data definitions for ATCO.CIF
In contrast to ATCO.CIF is the data dictionary for DATEX. A table included in the appendix includes the data definitions related to public transport (part 4 only). Comparison suggests that it is similar. To compound this, it will be seen though that most of the data definitions in DATEX correspond to Transmodel but still concentrate on what the operator needs, and not on what the passenger needs. It is important to make the link here that data definitions can be related exactly to “keywords” in the later messaging structures (Chapter 4). Translation from one data dictionary to another is expected to be minimal as the elements in the DATEX and TCIP data dictionaries, in so far as could be determined, are broadly similar.

2.5.3.2 TCP ICE Definitions

On the other side of the Atlantic, the Americans decided on a similar set of definitions.

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stop Point</td>
<td>Description of a point where transit customers board or alight from a transit vehicle in revenue service.</td>
</tr>
<tr>
<td>Block</td>
<td>See Vehicle Assignment.</td>
</tr>
<tr>
<td>Block Group</td>
<td>A grouping of blocks based on some common characteristics such as use common corridor, terminus location or route direction name.</td>
</tr>
<tr>
<td>Driver Activation</td>
<td>A specific request for a driver to take an action. This request is a type of event.</td>
</tr>
<tr>
<td>Event</td>
<td>An action which is activated at a specified time, location or both. Examples of events include: announcements, changes to route and schedule adherence requirements, changes to variable message signs, request for a driver to take some action, notification of a change of a fare or radio zone, and notification of a change in operator (relief driver).</td>
</tr>
<tr>
<td>Master Schedule</td>
<td>A table that includes all the time points and trips on a route. Contained within the route description is the Master Schedule Header information.</td>
</tr>
<tr>
<td>Master Schedule Header</td>
<td>Sequence of time point identifiers and/or their names used to define the order of time points for all the patterns. This field is used to build timetables, for exterior signs, etc.</td>
</tr>
<tr>
<td>Operator Assignment</td>
<td>The daily pieces of work assigned to a transit employee.</td>
</tr>
<tr>
<td>Organizational Unit</td>
<td>A division or group within a transit agency, for example, Operations, Communications, South Garage.</td>
</tr>
<tr>
<td>Pattern</td>
<td>One of multiple outer route segments served by a single transit route.</td>
</tr>
<tr>
<td>Piece of Work</td>
<td>A task or series of tasks that, when carried out, provide the basic components or building blocks of transit service delivery.</td>
</tr>
<tr>
<td>Public Transit Vehicle</td>
<td>Vehicles owned, leased or subcontracted by a public transit agency to support service provision.</td>
</tr>
<tr>
<td>Roster</td>
<td>Daily operator assignments grouped into weekly assignment packages.</td>
</tr>
<tr>
<td>Route</td>
<td>A collection of patterns in revenue service.</td>
</tr>
<tr>
<td>Run</td>
<td>See Operator Assignment.</td>
</tr>
<tr>
<td>Schedule Time</td>
<td>The time used to define a schedule day. A schedule day spans 48 hours in seconds [-43,2000, 129,599]; the day extends from noon the day before the schedule day until noon the following day.</td>
</tr>
<tr>
<td>Time Point</td>
<td>A geographic location which a transit agency uses to schedule transit service and monitor adherence to the service schedule. It is a point at which time is assigned to create trips.</td>
</tr>
<tr>
<td>Transfer Cluster</td>
<td>A collection of stop points wherein transfer between routes is valid and possible.</td>
</tr>
</tbody>
</table>
Trip | A one-way movement of a transit vehicle in revenue service between two points.
---|---
Trip Time Point | A specific time assigned to a timepoint on a trip.
Vehicle Assignment | The work of a vehicle from the time it leaves the vehicle base until its next return.

Table 9: Transit Communications Protocol definitions (ST-ITS-TCIP-SCH, Feb 1998)

In the end, an intersection of the essential elements of ATCO.CIF and those determined from the journey planner survey were adopted. However, the final issue of the protocol may attempt to make a compromise between existing systems in so much as will be possible, given the hindsight that time has afforded.

2.6 Survey of Journey Planning Resources

A previous section of this thesis discussed a comprehensive reference compiled by Casey (1998) for the USDOT, describing the State of the Art for Advanced Public Transportation Systems, i.e. a list of PTI resources distributed throughout the World Wide Web. For the research in this thesis, an ad-hoc, independent survey was conducted some years before this document was published (the sites investigated are listed in Appendix B). However, in the months since this thesis was written, the number of available sites continues to increase. Generally, they point to a common set of issues.

The survey of electronic journey planners enabled a number of important trends to be readily determined. For instance, many of the resources discussed in the introduction or referenced earlier developed within six months to a year. The trend was that the technology developed quickly until it reached a limitation (boundary, mutual interrogation, and multi-modal), and then stopped. For this research, it showed with a limited degree of confidence that journey planners had similar functions. To provide similar functions, it seems obvious that journey planners must have similar components. This limited survey also showed that journey planners had similar input requirements or parameters. It suggested that journey planners had similar outputs. By virtue of these similarities, the data requirements internally to these journey planners must be similar also, and the databases of timetable data were likely to be composed of similar data also.

2.6.1 Functions of Electronic Journey Planners

Journey planning has two primary functions. The first function is to build or construct a journey definition; that is, to define the parameters for a journey. The second function is to find a solution to the given journey definition. Simply routeing requires the determination of a selection of feasible routes between the origin and destination, via a sequence of intermediate exchange points, as necessary.

At least three methods of defining parameters exist. One method is to match characters entered by the traveller. A second method is to gradually refine the traveller's parameter through providing subsequent lists of options. A third method is to allow the traveller to use a pointing device and gradually refine the
traveller’s choice through a series of maps related with co-ordinates. This is discussed further in Section 2.6.3 and Chapter Five.

2.6.2 Components of a Journey Planner

It is necessary to break up a ‘journey planner’ into several components according to function. This section examines what components make up a journey planner and the functions each performs. For methodologies to achieve this, one can reference theories by Ford and Taylor (O’Conner, 1994) e.g. dividing a task to be executed into its smallest components.

By examining existing journey planners available at the onset of this research, it became apparent that all journey planners have several common functions. They display fields called text boxes within which the user can input data. They take user input from these fields and use that input to query the data. They perform calculations on the query(ies) and make judgements based on presumptions or user criteria. They display the results of these judgements.

It was useful to categorise the functions of a journey planner into several components. This research identified six components (capital C in following list), called either services or engines. Data is exchanged between the six major components:

C1) a Traveller and a software program—connects the traveller to a collection of components which together act as a journey planning service.

C2) the Presentation Engine—collects and forwards data from the traveller, C1; exchanges data with C3 or C5, and returns formatted data back to the traveller, C1.

C3) the Enquiry Engine—collects and exchanges data from the traveller via C2 and queries C4 to find out where C6 is so as to forward and collect responses;

C4) a Reference Table—(usually implicit and not well referenced) it relates relevant data in C5 to where C5 is physically located, in response to a query from C3;

C5) a Journey Routeing Engine—analyses data forwarded from the C3 component, searches in C4 to determine what data is relevant and where the data is located, requests data from a C6 components, receives

![Figure 2: Journey Planner Components](image)
the responses, analyses them, and then forwards a collated, comprehensive response to the C2 component.

C6) the Data Services--store and provide the data.

Research studies by McCormick (1994) and Smith (2000) identified four components (C1 and C2 as one component, C3 and C4 as a second, C5 as a third, and C6 as the fourth). The EUROSpin (Gilles, 1999) project shows a similar configuration as pictured in Figure 2, but as with the first two studies, it lacks the "reference table" or index. A fourth project (Brigges, 1999) seems to have identified the same components as this research, though existing documentation is not absolutely similar.

2.6.3 Criteria used for Journey Planning

To plan a journey before travelling, whether alone, assisted by a telephone enquiry service, computing machine, other communication, calculating machine, or some combination thereof, a selection of data is necessary to define parameters of the traveller's search. These data limit the scope of one's search.

These parameters tend to fall into location or temporal categories. An obvious example location that is applicable to all journeys is "What are the origin and destination of the journey?" A temporal example is "When does the traveller need [wish] to depart or arrive?" This parameter includes a date and a time.

Other factors of journey planning, such as price, environmental impact et al, depend on the travellers' preferences or affluence. But what preferences or services are available? What types of data affect a person's decisions? The following paragraphs refine the answers to these questions.

2.6.3.1 Defining the Origin and Destination

For travellers within the local area (a village, town, or smaller city), and some times within the local administrative area such as a county, the traveller typically just needs the origin, destination, date and time. The last two parameters, if not specified explicitly are normally interpreted as now.

In the case of a local journey to an adjacent or adjoining county, where an adjoining county is defined as a non-adjacent county that is joined by direct bus or coach route, the traveller may still only need to declare the origin, destination, date and time. However, as the traveller is crossing one or more administrative boundaries, some further details may be necessary, as discussed in greater detail in Chapter Five.

For longer journeys, either to a more distant, non-adjacent county or across the county or to another nation, the traveller will need to know: the origin, destination, date and time. Further, the traveller may be required to declare further details also. These may include any of the following:
• postcode (an address)
• geographical co-ordinates (commonly known as grid references in England)
• street name or junction
• a point of interest

One salient factor in the design of a journey planner system is to understand the sheer quantities of data involved, discussed earlier in section 2.4.2:

2.6.3.2 Defining the Date and Time of Departure or Arrival

Since it is possible to arrive on a different day than the day of departure, it is necessary to allow the traveller to force the system to search explicitly for options allowing this scenario to occur. Several date formats exist to cater for this.

Sun, 06 Nov 1994 08:49:37 GMT ; RFC 822, updated by RFC 1123
Sunday, 06-Nov-94 08:49:37 GMT ; RFC 850, obsolete by RFC 1036
Sun Nov 6 08:49:37 1994 ; ANSI C's asctime() format
Source: http://www.w3.org/Protocols/rfc2068/rfc2068

Optional data definitions are available but are not essential for a functional prototype.

2.6.3.3 Methods allowing parameter selection

At least three methods exist which can allow a traveller to choose an option. As with other aspects of improving the interface with respect to the user (section 2.2.4 and later in 2.6.6), the preference of one or more of these is a complementary area of research. As such, it is sufficient to provide several alternatives and briefly describe them, without going into further detail as to of which method or methods is 'best' for any particular scenario. These methods are listed as follows:

• Alpha-Numeric entry
• Select from a list
• Point and click on a image

Alpha-Numeric consists of a "text box", either on a "form", on a software program, or within a html "form" on a browser. Alternatively, it can just be a 'command-line' input. In the case of choosing an origin, the user might elect to type a few initial characters. A drop-down list or 'list box' allows a user to view one or more displayed options. These lists are usually obtained from a database table or query, and might include a list of dates starting from today. The user selects an option from the list by holding a mouse cursor over the option and clicking or via keyboard selection using the 'arrow', 'tab' and 'return' keys. A third method consists of allowing a user to point and click on an image of a map or similar symbolically linked image. The computer then responds as programmed depending on the location of the
click relative to the various polygons embedded within the image. An example of this would be present the user with various images of maps at increasing levels of resolution and allow him or her to gradually pinpoint an exact origin or destination point. Other options are available, but these issues are discussed more fully in literature by INFOPOLIS (1998-2000).

2.6.4 Hardware and Software in surveyed systems

Another aspect investigated in conducting surveys of journey planning systems was what sort of hardware and software were used by these journey planners. Of those investigated, most journey planners were based on either MS-DOS (Ordnance Survey and Southern Vectis as visited in 1996) or Windows operating system, either in 16 bit or 32 bit form (Hampshire and Buckinghamshire County Councils). Journey planners for London Transport (Coucher, 1995) and an experimental model funded by the EPSRC (McCormick, 1994) were based on a mainframe (UNIX) operating system. Although no journey planners for OS/2 (IBM) and System 7 (Apple's Macintosh) were reviewed, nothing in these operating systems prevents their use for this purpose.

Of the journey planners reviewed as listed in Appendix B, most were viewed through an Internet browser. This method allows the user to look at the journey planner independently of the base operating system. However, this did not by itself overcome limitations of mode or company. Thus, it is necessary to examine journey planners themselves in greater detail.

2.6.5 Enquiry and Search Engines

As discussed in the survey section above, a journey planning information system has two important sub components. These components are sometimes called the enquiry and the search engine. Their purpose was ambiguously defined as a black box which was responsible not only for defining journey enquiry, but also for searching the databases (whose terminology for the data structure was reviewed in the previous section, 2.4.3) for journey solutions. For the research described here it was necessary to separate the components of the black box into useful entities. However, whilst it might be considered worthwhile to examine the extraction of traveller information from the database source information, such an examination is not necessary as many proprietary software packages exist for the search engine. Whilst derivatives can be written using standard shortest path algorithms, a number of suppliers for data and journey planner engines for the local government bodies in the UK market are active.

As separate commercial bodies, each company has developed their own journey planner using different technologies. Personal visits suggested that as many different methods of implementation exist as there are technologies to offer, with the notable exception of OS️. This reinforced the desire for any extensible system to be designed in such a way so as not to be dependent on any particular software or hardware.

OS️ is an operating system by IBM that is widely used in commercial enterprises
Furthermore each data supplier uses a particular search process which is proprietary. Sharing these databases could compromise proprietary advantages.

<table>
<thead>
<tr>
<th>Supplier Company</th>
<th>Product</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anite</td>
<td>RouteWise</td>
</tr>
<tr>
<td>Pindar Routel</td>
<td>Travel Guide</td>
</tr>
<tr>
<td>WS Atkins</td>
<td>TRIPlanner</td>
</tr>
<tr>
<td>Software Logic</td>
<td>Station Master</td>
</tr>
<tr>
<td>Southern Vectis</td>
<td>Zephos</td>
</tr>
<tr>
<td>AVSI</td>
<td>National Express Journey Planner</td>
</tr>
<tr>
<td>Clarity, BR Business Unit (SEMA)</td>
<td>Railtrack Journey Planner</td>
</tr>
<tr>
<td>ICL (previously EDS)</td>
<td>Transport for London Journey Planner</td>
</tr>
<tr>
<td>Cap Gemini</td>
<td>Virgin Rail Journey Planner</td>
</tr>
<tr>
<td>Logan International</td>
<td>Scottish Journey Planner</td>
</tr>
<tr>
<td>MVA</td>
<td>JESS</td>
</tr>
<tr>
<td>Teleride</td>
<td>TeleRider</td>
</tr>
<tr>
<td>Represented by Railtrack</td>
<td>HAFAS</td>
</tr>
<tr>
<td>Represented by AIM</td>
<td>IPTIS</td>
</tr>
</tbody>
</table>

(Source: meetings with the various parties above)

Table 10: a listing of most Journey Planner Suppliers and their Products

As discussed in Section 2.5, some suppliers had begun to set up a file format to exchange common schedule data which would not compromise their search software (Ness, 1996) to overcome these issues of hardware or software differences. Yet even whilst some suppliers had begun these initiatives, it was suggested here that it could be inherently more sensible to exchange only the relevant portion of the timetable database to the 'foreign' system as and when needed. Conversely, each supplier would have to ship the entire contents of each of their databases to every other supplier. Whilst this exchange can be manageable for two suppliers, it becomes an obstacle as the number of suppliers increases, more importantly, it can be wasteful of transmission resources. As in Holland, no system to date allows remote, autonomous interrogation of foreign databases or journey planners (i.e. interconnection). This characteristic was thus seen as crucial for any level of success in the English privatised market.

With respect to the enquiry engine, this component remains ambiguous, with documentation reviewed to date (Eurospin, EU Spirit) glossing over the intricacies suggested in the first section of this chapter, which are re-examined in Chapter 5. Some work in anticipation of the PTI2000 initiative has been commissioned by some county councils (Dyson, 2000) for enquiry engines, but overlaps with the presentation of information and thus is not covered in detail in this thesis.

2.6.6 Presentation of Information

The different methods of presenting information to the traveller are not examined in this thesis in detail. The Institute of Logistics and Transport, formerly the Chartered Institute of Transport, has set up an accreditation panel in the United Kingdom, and several larger European projects such as INFOPOLIS.

2.7 Summary

Thus far, a minimum set of enquiry data parameters thought necessary to allow a system of computers to compute a reasonable journey with respect to time has been determined, as has the minimum set of response data parameters for the expected output. This enquiry data set includes an origin, a destination, a date, a time, and the type of journey request. The response data set includes the origin, destination, departure and arrival time and date for each leg, its mode, the service number, and the service operator. For this thesis, these form the user requirements. Fare information is excluded.

A number of services have been examined and whilst this review found that a number of services are on offer, a number of limitations remain. With specific reference to journey planning information, no journey planner product was comprehensive as outlined in the objectives.

Research into the available data sources has determined that existing data sources are neither integrated with other data sources, nor comprehensive in detail.

Examination of existing methods of data transfer and messaging suggest that they are efficient for the purposes they are designed for, but no method has the ability to allow mutual interrogation between journey planning software.

Whilst it could have possible to look at more complicated sets of parameters such as fare information, and what effect their inclusion might have on any one component, or on the system as a whole, insufficient data and resources existed in this research to permit its inclusion. Future journey planning systems may be required to provide fare information (i.e. Transport Direct), and individual systems such as TheTrainLine (Cap Gemini Ernst Young, 1999) already include limited features in this respect. However, a number of data and political obstacles preclude such an investigation.

In summary, this Chapter reviewed the travellers' requirements, existing innovations, projects, and other sources of information. Data inputs and outputs for the user were largely uniform, and the internal components of journey planners were similar, both in function and in order of flow. Current systems have natural obstacles in extending journey planning beyond software, geographic and database boundaries. Whilst an examination of existing data dictionaries found a sufficient degree of uniformity, some variance in the exact words and definitions caused inconsistency. Methods of data transfer did not offer mutual interrogation. Thus, these examinations of available data services and sources suggested that sufficient data existed in the public domain or within licensing agreements in digital form of sufficiently similar format (i.e. inside databases) to permit continuation of the study towards its objectives. Therefore,

---

4 Known research under investigation can be found at http://www.soton.ac.uk/~pti
the next Chapter continues the literature review by investigating philosophies in data storage and architecture. However, it is noted that pursuit of such an endeavour is risky due to the pace of technological innovation and a lack of a sound business model in a commercial, privatised, and deregulated environment (Wilkinson, 20-22).
Chapter 3 – Review of System Architectures

3.1 Background in Database Technology

It is necessary to introduce some background material before discussing and evaluating possible system architectures. This material includes a brief review of database technology as it has evolved, primarily with respect to public transport timetable publishing. Later sections build on this material by suggesting a particular architecture for an extensible journey planning information system.

Several digital technologies exist to store timetable/scheduling data. In the 'early' days of widespread adoption of digital technology (1990's), word processing and desktop publishing were the main tools used to convert traditional printing methods to digital timetables. This evolved somewhat into the use of spreadsheets and tables in a database as publishing companies began widespread adoption of inexpensive software. Alternatively, some suppliers wrote simple, customised databases to store their data.

Of the three digital technologies, the database proved to be effective for journey planning and is thus described in greater detail. A database could be described as a collection of information in an area. Data in a database is stored in records, where a record contains a row of fields. Fields are arranged in columns. Database software typically displays this in a tabular format, using a table as a collection of rows and columns. Generally, a reference or identifier is in one column, with data held in subsequent columns. These identifiers allow relationships between tables within the same database, and between tables of different databases. These relationships can either be defined explicitly in the beginning (a structured database management system) or through 'joining' data in two or more tables (a relational database management system).

Microsoft SQL Developer's Companion states that structured database management systems can also be known as network or hierarchical systems. Unlike a relational database, these data structures are predefined in an exact sense. For instance, the number of characters allowed in a text string is fixed rigidly, and often in such a manner as to minimise overall storage space in either RAM or on a fixed disk. Relationships are fixed also, and to change relationships would require a major modification in the structure of the database, which may lead to inefficiencies in database performance.

Either database management system can be appropriate. In structured systems, relationships between data are predefined. However, a relational database management system allows 'unanticipated' relationships between data. Thus, a degree of flexibility not available in structured database management system exists, which can be advantageous in the design stage, but as above, may lead to inefficiencies later. With respect to this research, it is more important to discuss any disadvantages that may exist in data maintenance. From this perspective, faster systems have been the trend, and often render the difference between an efficient and inefficient system meaningless. However, inaccurate data have a high cost regardless how efficient the system may be and can be difficult and time-consuming to remove.
Thus, when a database is held entirely within an organisation, the decision whether to hold the data in a structured or relational manner is largely a matter of their choosing. The research described in this thesis looks at crossing boundaries between organisations instead, so it is important to find a common ground regardless of the individual organisations' decisions.

As stated in the last chapter, County Councils typically tendered all the timetables for their area to one timetable database supplier, that is, to one organisation (Slevin, 1996). A council represents a specific geographic area, and neighbouring councils have similar data, but perhaps tendered to a different supplier. Should a traveller wish to travel out of the geographic area, it is necessary for them to determine their own way. Thus, a need arises for a third party to extract a small data slice from more than one database, and as such, difficulties arose, because inevitably, one company chose one operating system and one manner of database management, and the second company choose another manner.

To take a slice of data from a database, it is necessary to introduce the concept of a query. A query may be thought of as a specifically formatted question. A query takes an extract or slice of data from a much larger tract of data according to one or more criteria. For instance, in a list of villages within England, a query might only return the villages beginning with the letter "A". Alternatively, it might list a selection of bus stops within a specific distance of a rail or coach station. Thirdly, it is possible to return just the first trip that goes between one stop and another on a given day after a given hour.

By combining occurrences of the third possibility, it can be seen that a journey, defined in this thesis as a collection of trips, could be constructed. In this manner, the concept of a query is quite useful in journey planning. A further advantage of a query is that it is just a copy of the original data in the database, containing only the relevant data. As a matter of aside, the arrangement of these relationships affects the efficiency of the question or a 'query'. In journey planning, several methods exist to organise data into tables. But it is necessary now to discuss the fundamental architectures before going into further detail.

3.2 Basic Architectures

Following the above, philosophies in data storage can be considered under two basic 'architectures'. Traditional design had been to have everything in one place. The data can all be stored in one location, such as an office or building. This is the so-called "warehouse" concept. Alternatively, it can be thought of as a 'data reservoir' (Bell and Crimson, 1992). In this architecture, all the suppliers dump their data into a reservoir and all users get data from the reservoir.

An alternative architecture is a distributed design. In this design, data are spread out between various locations, say in several offices in various counties or cities around the country. Indexes relate the tables or databases together. This index might be analogous to relating a grid reference (Latitude and Longitude elsewhere) to a place-name as in the back of an atlas or on a map.
Some advantages and disadvantages can be illustrated using an academic example. Say for instance that a University has several thousand students and academics, which can be aggregated by Faculty and by classification. For argument's sake, presume that three Faculties exist in the University, and several global classifications exist: undergraduates and post-graduates on the one hand, and lecturers and professors on the other.

Presume for a moment that the University kept a universal, centralised and controlled system for keeping student data in a reservoir approach, and say that this happened to sit in one large faculty, which frequently used the data reservoir. Other faculties are small by comparison, but might require access just as frequently on a per unit basis. Assume also that there is a first-in, first-out queuing policy in place. By using a computed probability alone, it can be shown that members of the other faculties might incur an added 'cost' with respect to time in accessing the data reservoir.

If the University adopted an integrated, decentralised (distributed) database architecture instead, it would be possible to create three separate database reservoirs, linked by a small, fourth reservoir or index. The index might commonly be held by all reservoirs as an alternative. Whilst the access for each of the smaller faculties could improve in this instance, the duplication of the index might incur an overhead cost disproportionate to the benefit which arose from a reduced queuing time for access to the one reservoir.

The above illustration presumes autonomy of the organisation and that it exists in total isolation of all other factors. In expansion of the example, presume several Universities exist in limited competition with each other. These limitations are based on the presumption that a certain percentage of their students comes from the local area; a smaller percentage of students comes from farther away.

From the perspective of a national media reporter, this reporter might want some statistics regarding comprehensiveness or quality of one University to compare against another. The difficulty in getting the same statistic from both Universities, and having this statistic with the same definition and interpretation, is a challenge acknowledged in the National press (Guardian, 1999).

The following paragraphs transfer the analogy to an extensible journey planning system. Say that two suppliers of comprehensive database services for public transport establish and build their own suites of services in different areas of a country. At some stage a traveller will wish to cross the boundary from one area, controlled by company A, to another area, controlled by company B. The traveller does not care, nor should have to care, which company supplies him or her with the data about the journey. In a monopolistic environment, one company would have to conform to the practice of the other, as the first controls all aspects of the market. However, the traveller is sometimes disadvantaged by this situation as only one source of information exists, and thus invariably, some bias will exist.
To encourage integration between information services and discourage the prevalence of biased information, the Government has established more clearly in recent White Papers that the information not only should be unbiased, but must be unbiased. (DETRc, Sec. 8.7, 1999). This requirement effectively makes one national supplier of public transport information unlikely. An effect of this policy is to encourage competition in the publication of the schedule information also, and not just in the provision of transport services. This competition suggests that an additional layer to the data hierarchy should exist so as to link the different geographic areas. This additional layer would have to be different from some of the lower systems, and thus a hierarchical, distributed database management approach to an extensible, national system is suggested. In this way, councils could encourage competition for the renewal of the service, or for different quality levels of the service, or even the supply of the service within a particular area or (sub) region. However, if the overall number of users is small and the quantity of data also relatively small, the market-suggested solution may not be optimal.

To summarise, two architectures have been discussed in this section: a centralised architecture and a decentralised (distributed) architecture. From a resource perspective, both architectures have a fixed cost, and a fixed access time. Depending on the structure of the organisation, the number and frequency of users, and the frequency and number of suppliers, it can prove more advantageous for all parties involved to decentralise the database, whilst keeping an integrated database management system. With respect to a journey planning system, it is necessary to examine the related trade-offs in greater detail.

3.3 Design, Analysis and Evaluation

Several issues affecting the overall design of the architecture for this research model have been discussed thus far. The first was whether or not a structured database management system or a relational database management system was preferred. A second broke down the components of the journey planner system into specific functions. The third was whether the system should be centralised or decentralised. At this point it is useful to relate these before discussing the influence of distance on this system design.

3.3.1 Design Parameters

A key issue in the design of an extensible journey planning system is 'where the data is to be stored'. A second dilemma that arises is how to the data can be connected or 'related', as referred to above. So far the background material has suggested that, within a particular region or geographic area, it was not important whether the database management system was structured or relational. However, the volume of usage, both in data upkeep, i.e. input, (discussed next) and data supply, i.e. output, may encourage a particular architecture.

\[\text{In the license conscious culture of England, it was perhaps a more significant hurdle than might have been encountered elsewhere}\]
Using data discussed in the second chapter under data requirements, some rough estimates illustrate the scale of the design. As in the earlier chapter, two primary media channels exist for this public transport information: within a national network of call centres and via a national Internet. For the basis of illustration, assume 60 million enquiries by phone and 40 million enquiries by Internet (i.e. 20 times the annual enquiries in London, Bourne, 2000). The total of this equals 100 million enquiries per year.

Presuming that the call centres are inter-linked, and if by supposition there are 10 enquiry centres, then that leaves 10 million enquiries per year per centre, or roughly speaking 200,000 per week per centre, or 30,000 per day. Assuming that most enquiries occur during a band of 20 hours a day, the average of this is 1500 enquiries per hour, per centre. A peak hour service might be slightly more than twice the average, which, for the sake of rounding, might give 3600 enquiries per hour. So as an input, this design-exercise estimates that each centre might be expected to have a capacity of one enquiry per second. The call centre computer system is not concerned whether the enquiry is by phone or by Internet.

Assume each enquiry produces between one and 25 actions as a result of an enquiry. Using an 80/20 split, (i.e. the sum of one times 80 and 20 times 25, and the aggregation divided by 100) this results in a mean of about six (6) actions for each enquiry. If the average action time (time to respond by the computer) is 0.5 seconds, then the average response time for each enquiry might be three (3) seconds, and suggests an absolute minimum of six resource units, per call centre. Even with these volumes, the necessary computational power is hardly overwhelming. This does not justify a distributed approach.

Should the processing time by the call centre operator be 120 seconds per enquiry (Lynn, 1996), and the maximum calls per operator be ten (10) per hour (British Gas, 1998), with one computer per operator, the quantity of idle computation time (about 90%) is significantly greater than the active time (about 10%). This result is due to the system being constrained by the human productivity, and not the computers.

3.3.2 Data Maintenance, Data Accuracy, and Integrity

Within each database area, some data maintenance is required. Normally, this responsibility has fallen and remains with the council (DETRac, 1998). However, bus services change frequently and at no fixed interval; sometimes these changes are grouped together such as with bank holidays, sometimes they are weekly, and sometimes unpredictable, as for new services. The burden of upkeep can be quite large (GMPTE, 1999), i.e. around four fulltime employees per year for a database of 16,000 timing points.

Each and every area administrator wishes to reduce this burden of constantly updating the database, and the resultant need to republish new timetables of services. Efforts from within the industry have then sought to transfer some or as much of this burden as possible onto the transport operators for a local area or region through a process know as "Electronic Bus registration" (DETRg, 2000), briefly discussed in Chapter Two. Meanwhile, it is unlikely that any one administrator would take on this burden of management at a national level. However, under a distributed, integrated system design the cost of maintaining the data can be distributed also.
By keeping the burden of database maintenance local, data accuracy is maintained. It should be obvious that if a centralised database update centre was established, as at least one company has tried to do (Lynn, 1996) in its timetable for Great Britain, the accuracy of the data becomes increasingly more suspect (Funnell, 1998). Thus, local geographic knowledge is essential for high levels of accuracy for public transport journey data. Also, the cost of data maintenance is spread, so whilst higher overall, is not any more expensive than the current situation. Therefore, accuracy was more important is so far as data maintenance would not cost any more, and any savings achieved from a centralised data maintenance was likely to make data so inaccurate as to be useless in practical circumstances.

Integrity of a database is also important, and relates to the consistency, validity, and accuracy (as above). Bell and Grimson (1992) state that database integrity is "concerned with whether or not the state of the database obeys the rules of the organisation it models". Mainly this is concerned with mistakes caused by incorrect or inconsistent data input, or invalid output from executed computations. It is easier to maintain higher integrity in a smaller database than within one with millions of individual and often manually input data fields. This statement suggests that it would be advantageous to link and index hundreds of small databases with higher data integrity than try to merge these all into one or more larger regional or national databases.

3.3.3 Security

In a national context, many foreign, incompatible, yet similar proprietary database systems exist which are updated frequently. Trying to import each selection of database updates into a single large database is expected to lead to higher possibility of a breach of security. High security could be maintained within each of the smaller databases by placing protection around it. The database then distributes requested information as needed, saving duplication as well. More information can be found from references available in Grimson and Bell (1992) and similar texts. Later research such as the more commercially oriented EU-Spirit is expected to concentrate on this aspect in greater detail.

3.3.4 Extensibility

Both centralised and decentralised database approaches are extensible to varying degrees. With a warehouse approach, as each proprietary data supplier joins, new negotiations are necessary to allow the new entrant into the system. The distributed approach needs an agreed, established, public protocol for data exchange, with a well-defined set of common terms. In this approach, the new entrant need only adopt his proprietary system to the public standard. This flexibility allows the data operators to maintain their own portions of any national system, and it allows areas to come on-line in stages without affecting the existing availability. With a warehouse approach, this process is less transparent, but still possible.
Thus in the distributed approach, it is easy to add new entrants to the system, or to improve resolution or detail, as more county councils and others produce journey planners and comprehensive databases.

3.3.5 Redundancy

Both approaches allow for redundancy also. In a warehouse approach, all information enters and exits at one point. If any link to this point is severed, then all data are effectively lost. However, in a distributed system, it is possible to include more than one route between the two sources should any link fail. It is also possible to provide more than one data source, i.e. to "mirror" a data source inexpensively. This concept is sometimes called a "data farm". Should a source or even a whole "farm" fail for any reason, only data within that source/farm are affected, and not the entire system.

An advantage of a distributed approach is that secondary or backup systems can be added relatively easily. These secondary systems can be added or upgraded whilst a service is running, without taking down the service. Finally, different levels of redundancy, i.e. fail-safe, are enabled, and each proprietary system can independently decide what level of service they wish to offer. Thus, whilst possible, a warehouse approach would not be as flexible or perhaps even feasible in this environment without it mirroring the distributed approach.

3.4 Influence of Engine and Transmission Speed on Design

On the basis of the above, it was concluded that most of the components of the journey planner should be distributed for a regional or national system. It was necessary then to include in the design some method of data transport. To accommodate the different distribution channels, a common but flexible method for data transport was necessary. Several standards of data transport exist and each has their advantages and disadvantages. Of all known standards, only two are universal in that they allow different hardware and software systems to talk to one another. Of these two standards that exist globally, TCP/IP has become the de facto or default standard. Within TCP/IP, several standards exist, although this analysis is left until the next Chapter whilst the influence of transmission speed on location is examined.

Having resolved that a mostly distributed architecture was preferred, the next logical question was to examine if it should be a more-distributed-than-not architecture or a less-distributed-than-not solution. Performance measurements may only be undertaken after the concepts have been proved in a prototype system; thus apart from the overview discussed here, further discussion is continued in Chapter Seven.

Experience in the financial sector had shown that it should be a more-distributed-than-not architecture if accuracy was to be an over-ruling factor. However, it is worth investigating the possibility of a "rule of thumb" measurement of the relative importance of distance on individual search response time, from which later research could prove or disprove if necessary.
3.4.1 Theoretical Internet Speed

The primary advantage of using the Internet as a transport medium for data exchange is that small quantities (or large quantities) of text in a message can be transported large distances (ntl, 1999) at the speed of light (Giancoli, 1989) as a result of Maxwell's equations. For illustration, this research examines the influence of transmitting a 10,000 byte message (ten times the sample transmission in Appendix E) at 256mps for 1,000,000 metres, or about Aberdeen to Land's End.

This time can be computed as the distance divided by the speed of light, which results in seconds, plus the message length divided by the frequency. In this example, this would be computed as taking approximately 0.003 seconds plus 0.0004 seconds. Rounding generously this would take 0.004 seconds, as some delay would occur from passing through the repeaters and bridges of the network. Larger messages (25 MB) would take about a second under these conditions, assuming no other traffic. Special characters such as the Chinese character set require more bits (i.e. UNICODE would require 16 bits), effectively halving transmission rate, or doubling the time.

However, true speed outside a laboratory environment would require more detailed examination. Halsall (p 55, 1996) states that transmission propagation reduces the signal speed in typical transmission mediums such as twisted pair and coaxial cables by a third. Whilst this is not significant in a laboratory setting, nor perhaps even if isolated to Wales or England. However, due to the larger distances encountered in the North, it could be significant in signals travelling from southern England to Scotland. As data is normally transmitted in blocks (also known as frames), Halsall suggests a further delay associated with the acknowledgement receipt that is returned to the sender. Therefore, whilst not measurable in a single laboratory environment without hundreds of kilometres of wire and numerous routers, their combined impact in a real implementation could be significant.

Furthermore, the system is constrained by the transmission speed in some cases, as many of the connections to the backbone network in England are still at 64 Kbits per second or less. Thus the message size is important in so far as the branches of the network are concerned. This is discussed later in Chapter Four.

Meanwhile, just as in motorways and road networks, too much traffic causes congestion. The same is true on a network cable or leased line. If the volume of transmitted data exceeds 50% of design capacity in some mediums, then throughput suffers considerably just as in road traffic theory. In a packet switched arrangement, the traffic need not all originate from the same source. An analogy is a motorway which has significant feeder traffic. Another analogy is if the motorway has significant numbers of exits and entrances over a short distance; this too can reduce throughput. Finally, different mediums have different safety margins, so a token ring network performs better than a twisted pair. Price of the relative options

---

6 Actual transmission speed is different than theoretical transmission speed as the delay of passing through multiple routers can be significant, as can handshaking (See Chapter Seven).
with respect to throughput is a factor commercially as the token ring is more expensive. (Halsall, p184-216 an 271-343, 1999)

Whilst data compression techniques can lead to reduced quantities of data transmission, the scope for this is limited by application. For instance, in video transmission, transmission of every other bit would reduce the quality of the image, but it still may be recognisable at the other end. On the other hand, not transmitting every character would result in gibberish to the human at the receiving end. However, if there are patterns or repeated words or bytes in the data to be transmitted, some reduction is still possible. (Halsall, 137-155, 1999).

In summary, a number of factors influence transmission speed. Whilst a number of these could substantially affect the overall performance, in a commercial implementation they are largely a function of price and existing capacity. In the prototype for this research, testing these factors in addition would require significantly more resources than were available.

3.4.2 Theoretical Engine Speed

Whilst it would be a rather difficult task now, thirty years ago one could measure the expected time a query calculation would take based on various factors. Older design manuals sometimes talk about latency and workload. These factors are nearly meaningless now. ^ (IBM, 1996)

The restraining factor in the design of a national system is expected to be the search engine. Response times from any engine are expected to range from under a second to as high as a minute or more (Ness, Houghton, Marshall, and other engine suppliers). To a limited degree, the engines’ response time are a function of the size of the database that they search, that is, the absolute number of records. As the number of records increases, the search time will increase, but not at the same rate with all search engines.

For instance, if a computer can handle 5,000 computations a second, and if a binary search method is used, then the average time to find the n\textsuperscript{th} record in a set of 10,000 records is expected to fraction of a second. In this method, two computations are necessary for two records, three computations for four records, and so forth till fifteen (15) computations searches 16,384 records. Thus if a computer could handle 1000 computations at 33 MHz, and 10,000 at 750 MHz, then that equals a significant improvement at 2\textsuperscript{5}. It is relevant that processor speed has increased 25 fold (from 33 MHz to 750MHz) in the last five years, and doubled again to 1.5 GHz in the last year. Whilst this improvement is not absolute with respect to computations, it has meant several orders of magnitude improvement over the last few decades in hardware technology.

Software technology has not improved so greatly. This fact can be illustrated by comparing the same query on the same data set using both MS SQL and ORACLE. Both database software programs give quite different results with respect to speed; this is true also as the database size is increased.

3.4.3 Design Decisions

It has been suggested that the UK rail network, the National Express coach network, and a rural county bus network are similar in size. (Houghton, Ness, Marshall, 1996-9) Each may have approximately 2-3000 nodes (depending on internal granularity such as including intermediate stops or just timing points) and databases in the range of five to fifty megabytes depending on size, supplier, compaction, and other criteria (Houghton, Ness, Marshall, 1996-9). Searches of these databases result in response times that are expected to vary between one second to one minute. The main factors giving rise to this variance could be from differences in relationships between the tables, location of tables within the hardware or software, the algorithm of searching done by the engines, the particular software used (NT vs UNIX), parameters chosen, and processor speed. (Houghton, Ness and Marshall, 1996-9).

Some suppliers suggest (Ness and Paylor, 1998/9) that it is still feasible to put 10 such databases together and still achieve reasonable response times. However, should 100 of these databases be combined into a single, continuous database, it is expected to have an appreciable, detrimental impact on response times. With nearly 100 counties in England and Wales, were one to combine all of the databases together, this could result in excess of two million nodes, just for the public transport network. (Subsequently, the size of the database might be expected to increase 100x or more). Searching on this database could prove time consuming.

If a large database was divided into one hundred pieces and search first the index and then the appropriate or relevant pieces, this is expected to exhibit a quicker response than a single, time-consuming search of the whole database. Therefore, as internet overheads are expected to be minimal compared to search engine response times, it is hypothesised that it would be best, presuming such a system could be set up, to create an index of sorts and link all the databases together, rather than try to merge all the database. However, the relative magnitude of this is important, and this magnitude is examined in Chapter Seven. Further research could determine an optimal national design, as some regions’ databases are quite small and may be worth combining. Other databases such as Greater London, Greater Manchester and Birmingham, might be worth splitting into one or more databases. This might be particularly true if the databases do not contain as much detail as they could, especially with respect to travellers whom would need an enhanced level of information. This enhanced information would also be necessary for users with a different profile than the perception of the ‘normal’ user. For instance, the user may need to use a route with special access, (sound, sight, wheelchair/pram) or simply require more time than the usual five minutes to make a connection to an adjoining bay or platform.
3.5 Specific Architectures for Multiple Databases

The previous analyses argue the various benefits and shortfalls of the two main philosophies. This section examines the various methods of constructing a system of databases.

3.5.1 Alternative One

In this alternative, the architecture requires that each database is converted to a common database standard, which are then joined. This allows the journey planner to execute a single set of queries to find a solution between any two points in the system. This has the consequence of avoiding the need to search more than one database.

This design has several advantages and disadvantages. For a number of small databases all controlled owned by the same supplier, it can prove advantageous because everything is in one database. The drawback of this approach is that, in a multiple supplier market, all databases must be converted to a common standard and shipped to the appropriate consolidator. The common database can become quite large. Even with the most efficient search engines and the fastest machines, response time could suffer. This effect is only compounded by schedule data with higher resolution (i.e. more timing points and greater detail regarding each stop). Shipping whole databases every time an update occurs can be onerous, as updates happen frequently.

3.5.2 Alternative Two

This alternative is slightly different. Each database is converted to a common format as in the previous alternative. However, instead of then bolting each database together, a separate table known as an index relates the databases to one another, using internal lists of exchange points. In this approach, a journey planning software would presumably have to “broker” the incoming requests for services.

Consider for illustration the following scenario. For this particular example, the research will presume that the consolidated database area includes all data within the South East of England. Presume then that at least two requests have come into the journey planner. The first request for an area in database one (1), and the second request for a journey composed of journeys from databases two (2) and three (3).
In the event of the first request, the journey planning software would seek from the index table the relevant information, and determine that database one is the correct source for its query. In this case, the job of the "broker" is no different than in the Alternative One approach in the previous section. However, in not having to search the collective database, the response time of the search query may be quicker. It might be significantly quicker if the collective database is large relative to database one.

In the event of the second request, the journey planning software would seek from the index table the relevant information, but this time discover that the journey must be composed of information from more than one database. In this case, the job of the "broker" would be more complicated. It still might be quicker than searching the collective database.

The salient point in this approach is therefore that the consolidator would still have total control. In some instances though, some benefit might be achieved by searching a small index first, and then searching only the relevant database(s) for the information.

A drawback of this approach is that it requires more structure to the data, and as such would require more effort towards conformity. However, the efficiencies gained by the traveller in shorter response time might make this approach cost effective. On the other hand, the consolidator still has an effective monopoly, and to a user, any efficiency gained from a quicker methodology might be offset by the price charged for the service of passing through the consolidator.

3.5.3 Alternative Three

A third philosophy is much more distributed in nature. As depicted, several instances can exist of multiple brands of similar devices concurrently from the perspective of the traveller. The same is true of the journey planners or what is later labelled the public transport navigators. Finally, it is possible to have a proprietary method of searching for each of the data sources.

However, two commonalties are necessary. One is a common element containing a list of data sources to index against; this is described in greater detail in later chapters. It is also necessary to determine a common, impartial set of the messages and content, which the output of existing databases in the system can together subscribe to. In example, it is necessary to agree a common method of exchanging just part of the database as and when it is required.
The drawback of this approach is that it requires a great deal of co-operation. On the other hand, it allows each database supplier to maintain the internal integrity and confidentiality of his or her system.

The EU-SPIRIT (1999) consortium appears to have adopted this approach, some two years after initial papers from this research.

![Diagram: Generalised Public Transport Information System]

Figure 5: Generalised Public Transport Information System

3.6 Analysis of Alternatives: Chosen Design

The third solution described above was chosen as a basis for this research, even given that it would require a great deal of co-operation between database owners and thus a high level of risk. This approach might be labelled the more-distributed-than-not philosophy. Clearly, the more data that is obtained and updated at source, the higher the integrity of the data will be, and only that small quantity of data retrieved as necessary. However, it may yet prove to be the most optimal solution technically.

The research described in this thesis set out to examine this particular architecture, knowing it may not be optimal. Not having previous work to compare against (the EU-SPIRIT architecture was not published until 1999), it was felt best to pioneer a solution, and let further research determine if it was optimal or not. The EU-SPIRIT architecture suggests a design that is remarkably similar.
Figure 6: a model of a Journey Planning System

Therefore, the enquiry engine component interacts with (potential) travellers, often via the presentation engine, and assists them to define the basic trip definition of origin, destination, date, time, and departure or arrival by using a gazetteer. The basic trip definition is then forwarded to the routeing engine. The routeing engine uses the registry to break up the journey, determine who holds appropriate data on journey services and relate exchange points. The routeing engine component then formulates appropriate questions for data in a standard format to the respective remote routeing engines, interprets and analyses the responses, and collates appropriate portions of the responses. Remote routeing engines access data from databases via remote data services' interfaces. The presentation engine then formats the filtered responses to allow a meaningful presentation of the possible solutions to the traveller. Finally, travellers interact with these components via a software program (often resident on the local machine: i.e. a browser) which connects the traveller on-line or via an interactive Internet interface.

3.6.1 Flow Diagram

In a distributed database system with multiple suppliers, it is likely encounter incompatibilities between existing journey planner systems. However, as the data exchanged is largely similar in character, it is reasonable to presume that, given some incentive to exchange data, all parties might possibly commit themselves to a common system. However, as systems are proprietary, it is necessary to devise a method to allow system operators to maintain individual autonomy, responsibility and so forth, yet still participate in the mutual exchange of data. In other sectors, particularly the brokerage and banking sector, such a practice of arbitrage (via a broker) is well established. The principles of arbitration are that of exchange of data via a trusted "third party".
In computer terminology, the analogy of these principles of arbitrage is known as "handshaking", which then establishes a common, communications "protocol". Simply, all parties convert the data in their proprietary internal systems into a common standard. This is then exchanged, and the exchanged data is converted back into the proprietary internal system. The following paragraphs demonstrate how this "protocol" could fit within a "system" of journey planner components. Figure 7 shows the methodology of data exchange between journey planner components. Four basic objects exchange strings of data with each other via standard communication protocols. However, it is the two centre objects, the journey planner interfaces, which are the primary concern. These two objects "handshake" the protocols to be mutually agreed by the participants. All parties then are expected to adopt the protocol and at this stage in the development, it is released into the Public Domain.

Figure 7: Flow diagram of data exchange between components of the journey planning system
One object or application is the browser, which allows a conversation between the traveller and the journey planner, the next application. The protocol exchanged between these two applications is proprietary or copyrighted. The journey planner receives data from the browser and communicates with a JP or back-end interface normally of the Common Gateway Interface specification. The JP or back-end interface spawns “child” processes which exchange data with remote back-end interfaces. These objects then exchange data via the mutually agreed JourneyWeb protocol. The only difference between these two objects, labelled “JP-interface”, is whether or not the object is receiving a data string or sending it.

3.7 Summary

Two main philosophies of data storage exist. In the first, i.e. the warehouse, all the data is stored in one place, whilst in the second, i.e. the distributed approach, only what data needed locally is stored and everything else is obtained "at source", that is, from where it is stored remotely.

An advantage of the first method comes from a management perspective. In this philosophy, very high data integrity can be maintained at what is arguably a “minimal cost”. Access, data protection, backup, and other maintenance functions can be controlled from one location. However, this monolithic philosophy would require a great deal of database replication. The disadvantages of transferring large quantities of data are substantial, including whole databases on a regular basis, to and from a central store. Further, problems of database synchronisation become an added issue.

With the introduction of the Internet, a more flexible philosophy has arisen. This philosophy encourages the data to remain where it is best maintained (i.e. locally using local knowledge), and where it can be accessed on demand by the remote party as necessary and according to the appropriate rights and restrictions.

The traditional design has several advantages. If any part of this data is needed then, it can be shipped from a central location as needed. An arguable advantage of this approach is that all the data is translated into a private, internal system. Therefore it can be tightly controlled and licensed, with limits on distribution such as who access it and when it is used. This has the effect of centralised administration, which, from the perspective of the supplier, keeps costs down. However, being centrally administered, it also tends to mean that one organisation has overall control. This level of control may limit competition and, from the perspective of the traveller, either prevent access altogether, limit access to something of little value, or demand a high price for something which may have negligible value. For instance, it is not likely that a traveller will wish to pay 50 pence for journey planning information regarding a ticket whose price is only 70 pence anyway.

Another argument sometimes used to promote the data warehouse approach is that of higher data integrity, i.e. fewer errors, can be achieved. Also, the distribution of data and its replication are less of an issue. Whilst in a totally local system it is conceivable that higher data integrity is achieved, it is difficult
to imagine that this integrity could be maintained in a national or international area network if for no other reason than accuracy of base data. Furthermore, it can be argued both ways that extensibility is limited in this approach. Distribution, validation and replication of data from a local perspective are also easy to manage. However, to extend such a system nationally or further would require the warehouse to mimic the more distributed philosophy.

Thus, some manner of index is necessary at a national level, which will reference the geographic region or area to the supplier(s) of the data for that area. This index is sometimes referred to as a 'membership' or 'supplier list' within the industry. Whilst this is not explicitly necessary in traditional design, suppliers are increasingly adopting index principals in-house just to manage the huge volumes of schedule data (Houghton, Ness and Marshall, 1996-9).

However, it would not be appropriate for any administrator of a region or area, or a supplier on their behalf, to bear the cost of a national list, whose purpose would be to index the availability of resources one level below at the county or unitary authority level. That is, no authority should pay for more than their designated area, nor require any operator to pay for the whole of the national index on their behalf. The result, with such a large amount of data and numerous sources of data, is to distribute the cost of the national overheads geographically, with each area contributing a portion towards the costs of a national index.

To maintain autonomy or privacy of individual suppliers' systems, the data is accessed via a standard, agreed protocol. In this way, systems can then exchange and mutually interrogate data without disclosing the proprietary characteristics of the systems behind the protocol, giving hardware and software independence. As the search engine is the slowest link in the system, little impact is made on the overall performance using these designs.

Different companies adopt different concepts according to their needs. As illustration of this, even years after the introduction and wide-spread adoption of internet concepts, representatives from major companies are still making statements that all the data should be kept in one place. (Henrig, 1999)
Chapter 4 – Review of Data Messaging

4.1 Background of Data Exchange

This chapter introduces data transport and exchange, and looks at messaging structures and discussing existing technical barriers in detail. Later sections relate messaging and transport to the concepts of databases and transport information discussed in previous chapters.

4.1.1 Data transport and the Internet

A number of individuals, under the umbrella of the International Standards Organisation, developed the Open Systems Interconnect Reference Model as a basis for data transport to facilitate data exchange between computers. This reference model has contains seven layers that "define the functions of data communications protocols." (Hunt, 1992) The seven layers are Applications, Presentation, Session, Transport, Network, Data Link, and Physical (Halsall, 1995). Often it is pictured as several blocks, one on top of the other and is thus referenced as a protocol stack.

The Application Layer consists of the application processes which have the need to communicate with others. The Presentation Layer contains instructions that enable a standardised presentation to the user. The Session Layer manages communications between applications, whilst the Transport Layer detects and corrects errors in transmission. The Network Layer manages connections across the network for upper layers, and the Data Link Layer ensures reliable data across a physical link. Finally, the Physical Layer defines the specific characteristics of the network media (i.e. the cables). (Hunt, 1992), (Halsall, 1995).

However, the research in this study is based on the protocol architecture used for data communication across the Internet, known as TCP/IP. Descriptions of this architecture vary, but Hunt (1992) uses a four-layered approach consisting of the Application Layer, a Host-to-Host Layer, an Internet Layer, and a Network Access Layer. The Application Layer is essentially the same. The Host-to-Host Layer has a similar role to the combined role of the Session Layer and the Presentation Layer. Whilst the Transport Layer is analogous to the Internet Layer, where the data is encapsulated into a datagram and is routed to its destination. Finally, the Network Access Layer is analogous to the functions of the Physical Layer, the Data Link Layer and the Network Layer (Hunt, 1992).

Using the Internet as a basis for the messaging has a number of benefits. The Internet was created as a National Defence Information System capable of surviving even if large chunks of the network suddenly disappeared due to catastrophes such as nuclear war or simple mistakes such as someone cutting a cable. With this as a design philosophy, any message can reach any other point through a multitude of paths, i.e. if a particular path is severed, the message can still reach the destination through any number of
alternative paths. A second design advantage of the Internet is that the method of messaging is independent of computer software and hardware manufacturer, e.g. IBM, Macintosh, Microsoft, and Oracle. Thus any message can reach a destination even if bits of the network suddenly disappear, and regardless of computer manufacturer or software platform.

4.1.2 Background on Capacity

In a distributed system, two issues are important. One issue is the location of the data. The second issue is the capacity of the physical medium that connects data together. For example, storage of data is constrained by the space available on a data storage device such as a hard disk, though if the data is compressed (Halsall, 1995), the quantity of data that can be stored increases. The capacity of a physical network is limited by the ability of that wire, for instance, to transmit pulses of electricity without errors. An example of an error is a pulse whose positive voltage does not consistently deviate from neutral for a specific period of time.

![Figure 8: Sample Network](image)

As the Internet is an "always-on" physical network with intermittent packets sent across it, two packets may wish to travel along the same link in the physical network at the same time. Presume for instance that the medium and distance of all links in the network of Figure 8 are homogeneous, except that link CD has twice the capacity of the others. Then, if node A sends a message to node F at the same time as node B sends a message to node E, i.e. concurrently, it is possible that both messages will reach their destination without delay. However, if nodes A, B, and G send a message to F at the same time, one of the messages will have to wait, just as with a water main or pipe with a fixed capacity. As more water seeks to enter the pipe, the flow rate is (broadly) unaffected until the pipe reaches capacity. Once capacity is reached however, water collects around the entrance until the flow into the pipe decreases below the capacity of the pipe or an alternative route is found. It is important to note an obvious fact; it is cheaper to lease part of the capacity of a shared link such as CD than it is to obtain a dedicated line.

4.1.3 Quantity of Data for Transmission

An Internet 'pipe' is constrained by the frequency of the signal, the need for error checking, and a number of other factors. Taking a simplistic view, the speed is typically described in bits per second, and the actual capacity to carry data might be around half the stated speed (Halshall, 1995).
Application processes also constrain response speeds. When a request enters an application process (i.e. a web server), it is called a “hit”. This ‘server’ then processes the request in any number of ways. It might forward the request to another application process. It might make an immediate response. Finally, it initiate several requests of other systems, wait for the responses, 'splice' relevant information from the responses and collate them before making its “response” to the original request. Regardless, it needs parse the request; that is, to search a series of characters for a particular character or symbol which separates instances of "data", and then act accordingly.

With respect to the research for this thesis, the quantity of data exchanged is quite small. The data contain the particulars for a service, and may simply contain a line of text, which is defined in computer terms as a text string or simply, a string. For instance, the bolded, vertical column in Figure 3 shows the details of the number one (1) bus service, between the Bletchley Bus Station and the M.Keynes Shopping Centre. This service departs at 6:56 in the morning and operates only Monday to Friday (Not Saturdays) as operated by Milton Keynes Bus Company. This description consists of about 40 characters in a text string. Thus, the number of bits necessary to transmit this information would be 400 bits plus the header information, presuming each character required one byte (two for Unicode) and each byte has eight bits plus an overhead of two bits. Relative to the network capacity of the academic network, (previously four megabits/second, i.e. a paired set of “El’s”), the network time is not expected to account for a large percentage of any potential response time. It is obvious that if the network connection is dependent on a 2400 baud modem as was State of the Art in 1994, then the time to transmit the data would account for a much larger percentage of time. Relative to higher figures stating the capacity of network transport at about four million bits/sec, the number of characters or the physical size of the text string or “message” is very small.

4.1.4 Analysis of Internet protocols

Having suggested that data exchange would use the Internet Protocol (IP), it is necessary to decide whether to use the Transmission Control Protocol (TCP) Internet Protocol or the User Datagram Protocol (UDP) Internet Protocol. This decision depends on the size of the quantity of data be exchanged and the
reliability. UDP/IP is more suited to data exchange where the datagram is always small and where reliability is not an important issue, as no techniques exist in the protocol to verify "that the data reached the other end of the network correctly" (Hunt, 1992). For instance, if the datagram does not arrive at the destination, it could be re-sent. If it were always the case that only one service or a small list of names were to be exchanged over the network, this case would indicate that UDP would be acceptable, so long as sufficient time exists to re-send the data should the data not arrive at the destination.

However, it is still unknown how much information would be transmitted at any one time. If for instance, the quantity of information is large, say a whole day's schedule, then this might exceed a hundred journeys along the same route during the course of a day. If so, then it may be necessary to transmit in excess of 40,000 bits. As using a 2400 baud system, this transmission could take several seconds. Should it fail, then it would be necessary to send the data again, doubling the time. Given that a several of these transmissions are necessary to build up a potential journey, then several seconds could quickly exceed the eight seconds suggested in section 2.2.4. Given that the quantity of data could be large, and thus require it to be sent reliably, it is concluded that the TCP/IP protocol should be used.

Even so, several different methods exist within TCP/IP. Besides issues of speed and reliability, which are examined and tested later, a degree of functionality is also important. For instance, given the need to grasp an item, any pair of pliers will do. Should it be necessary to grasp a live electrical wire of high voltage, a pair of pliers with plastic handles is perhaps better. By analogy, it has already been alluded in the objectives that one system would have to interrogate other systems. An application process should have the ability to receive a string, analyse it, perhaps change it, and return it (the functions of a journey planner - see 2.6). Also, to overcome obstacles of several suppliers and a number of different hardware/software systems, and perhaps that of transport mode, it was necessary that all systems commonly or widely support the chosen protocol (i.e. easy to learn or implement).

4.1.5 TCP/IP based methods of Data Exchange

The choice of TCP/IP enables a distributed architecture, a heterogeneous collection of systems, and the reliable delivery of large quantities of data quickly. However, it is necessary to examine and decide which of the TCP/IP based methods of data exchange allows mutual interrogation between application processes. Ideally, flexibility in presentation to the user would be an advantage also, but it is not critical. Some flexibility in naming convention may also be critical.

Given these criteria then, several methods or protocols exist to exchange text data "packets" across the Internet (Krol, 1994). One method is Simple Mail Transfer Protocol, commonly known as Electronic Mail or simply, Email. A second is File Transfer Protocol. Two others are GOPHER and VERONICA. Another is Hypertext Transfer Protocol, which uses the Hyper Text Media Language and is commonly referred to as the World Wide Web (WWW).
Email is designed to allow a small ASCII text string, but may include a longer “attachment”. The attachment may be “UUENCODED” to allow it to pass images and non-standard characters as text. An email can be sent, received by a second machine, processed by the second machine. It can then send requests to other machines, wait for, receive, and process the responses, and then return a message to the original sender, in theory. Facilities for presentation are entirely text-based, but can be enhanced; generally, the method to enhance the presentation of messages is to send the text with HTML tags.

File Transfer Protocol allows files to be transferred across the Internet. Whilst it could be set up in the required manner, it is designed for a simple client-server relationship, and not a stacked system. Neither Gopher nor Veronica is widely supported any longer.

The World Wide Web (WWW) protocol allows the user to retrieve a text file, and depending on the arrangement of the characters within the text file, display it according to the parameters embedded within it. For instance, the text file may contain the string “<italics>some text</italics>”. This would cause the string “some text” to appear as “some text”, that is, in italics.

The WWW also allows any of the previous protocols to occur in the same software package, known as a browser. It functions by allowing users to follow links to files, often successively; this thus allows the user to browse the Web.

Hypertext Transfer Protocol (HTTP) and WWW have further advantages. One advantage is the product of linking text files in databases to software “scripts”, allowing the user to view a fresh page each time. This means the contents of the page can contain parameters whose values change with time. A second, more crucial advantage is the ability to “stack” CGI requests. For instance, one request can instigate an instance of a CGI script. This script then instigates one or more sub-requests on different machines, waits for and receives the responses, pastes or parses additional data as necessary, and finally responds to the original request. The importance of the realisation of this feature for public transport is a major contribution of this research. Discussion of the potential for this feature was discussed with an industrial partner in late 1995, giving impetus for further collaboration.

Another advantage allows a web page to automatically refresh itself at set intervals to allow viewing of changes in the data. An example of this feature allows a response to the user to indicate that further information is forthcoming. HTTP also has a point and click functionality, enabling a user to choose, for instance, a geographic co-ordinate on a graphical image of a map. Yet another feature enables streaming technologies, incorporating live sound and video files. Some of these streaming features are interactive; that is, they respond to actions of the user. Whilst the potential for these features is important, the ability to stack requests is the most relevant for this research study.

Whilst not an overriding point, it is relevant that in the development of web technologies (1997/8), a second interface, ISAPI, became available to co-exist with the Common Gateway Interface. ISAPI is a
proprietary standard of Microsoft, and is backward compatible with CGI. Ad-hoc trials of ISAPI did not show marked differences in response time or benefits in adoption. Thus this research used CGI.

4.2 Existing Message Standards

Having decided a combination of methods to ensure that the "data" could be sent over an internet, it is important to determine what methods exist to structure the data (i.e. the message) within the transmission.

At least two messaging standards exist. One is EDIFACT® (Electronic Data Interchange for Administration, Commerce, and Transport) which is an ISO standard sponsored by the United Nations. It has a number of standard messages and data elements, methods to add elements or segments, and methods to arbitrate differences. A second standard is ASN.1, which stands for Abstract Syntax Notation. Like EDIFACT, it aims to define common elements and segments for data exchange. A third messaging standard evolved from the DATEX consortium, and was described in Chapter Two.

A recent publication by the Trident consortium (Booth, J., 2000) suggest that EDIFACT is quite difficult to grasp quickly and is intended for the commonality of documents, whereas ASN.1 is more akin to the needs of computer languages and programming. It was not clear from DATEX documentation how the data is transmitted at the hardware or software level; only the data definitions are clear. For instance, it was not clear how the system architecture worked or what software might be used. It was clear that each data definition had a three letter "code", and most of the available documentation described these codes in detail. This problem was clearly evident in EDIFACT and TRANSMODEL too.

However, within DATEX (DETR, 1998), there exists a clear message structure. The message structure has three main parts: a salutation, a main body, and a closing. Another standard data messaging for the Ordnance Survey (Address Point) suggests this structure also (Ordnance Survey, 1999). As this methodology is established practice and meets all requirements, this research follows established procedures.

<table>
<thead>
<tr>
<th>GROUP MESSAGE HEADER</th>
<th>GROUP MESSAGE BODY</th>
<th>GROUP MESSAGE FOOTER</th>
</tr>
</thead>
</table>

Figure 10: Group Message Structure

The message structure of the protocol thus uses the following structure to embed sub-elements within elements. The salutation is called the group message header, with the main body being named the group message body. The group message footer follows as the closing. This structure is illustrated in Figure 10. Each of these parts is described more fully in the following paragraphs.

8 [http://www.unece.org/trade/untidid/welcome.htm](http://www.unece.org/trade/untidid/welcome.htm)
4.2.1 Header of Group Message

The header of the overall message needs to contain global characteristics of the message. These characteristics would include identifiers stating who is sending the message, their password or authorisation, a cross-reference, a statement of the number of elements contained within, and where to send the response.

4.2.2 Body

The body of the group message contains one or more messages or data strings to be exchanged as depicted in Figure 11. To insure integrity within the body, the protocol requires each message and message element to be separately delimited. Thus, each message has a message header, the body of the message, and message footer.

![Message Structure](image)

Figure 11: Message Structure

![Element Structure](image)

Figure 12: Element Structure

Within the message body, each element has an element header, followed by a value, and then finally the element footer as in Figure 12. For example, within a trip, an origin and destination both have dates and times which follow the same format. By adding this additional layer of delimiters, it is then possible for software programmes to differentiate between any dates or times associated with one or more origin and destination pairs. Otherwise, associated dates and times would only be two ambiguous data strings.

4.2.3 Footer of Group Message

The footer, or closing, terminates or ends the group message. The primary reason for closing a message is to insure message integrity. For instance, if a message does not contain the footer element, the recipient would have to request the sender to re-send the original message.
4.3 Obstacles preventing Data Exchange

Aside from the fundamental issue of data exchange, the interpretation of the exchanged data is also important. As the information may have the same meaning, but may be expressed in a slightly different manner, it is necessary to 'transcribe' it from one data supplier, council, or mode to another. Whilst humans can overcome this difficulty, it is not possible for a computer to overcome these idiosyncratic differences without explicit instructions.

4.3.1 Different Hardware and Operating systems

Chapter Two showed that several hardware systems are employed by data suppliers. In order to have an extensible journey planning system, the design has to incorporate a formula to allow software and hardware independence. At least five major types of hardware and software systems exist for desktop computer based systems.

- X86-based systems running Linux, OS, OS/2, (MS) DOS, Windows (3.1-2000) or Solaris
- RISC based systems, running Linux, UNIX or NT
- Macintosh based systems, running System 7 or later
- Personal Data Assistants, running Windows CE or other
- NetTV, running a version of Windows CE or other

Regardless of the particular manufacturer of the hardware or software in use, exchanging the data via the Internet allows almost total autonomy, as any of the above systems supports the Internet messaging standard, TCP/IP.

4.3.2 Different naming conventions

Fundamentally, this issue arises because each data supplier of transport information often uses a different name to describe the same physical point in space. The following table illustrates this in greater detail.

<table>
<thead>
<tr>
<th>Service Supplier</th>
<th>Ambiguous Example</th>
<th>Actual Place</th>
</tr>
</thead>
<tbody>
<tr>
<td>Railtrack</td>
<td>Bedford</td>
<td>Bedford Rail Station</td>
</tr>
<tr>
<td>Pindar</td>
<td>Bedford</td>
<td>Bedford Bus Station</td>
</tr>
<tr>
<td>Station Master</td>
<td>BEDFORD</td>
<td>BEDFORD Midland Rail Station</td>
</tr>
</tbody>
</table>

Table 11: Examples of different naming for a point

For instance, in the above example, Railtrack as the National Provider of information regarding rail services, states Bedford and by inference implicitly refers to the Bedford rail station. The data supplier company Pindar acting on instructions from Buckinghamshire County Council states Bedford and implicitly refers to the bus station, as the service describes bus transport and it thus refers to a roadside bus stop. Whilst Station Master contained mostly bus data as did Pindar, the data in this case referenced...
a rail service and thus the service stopped at Bedford Midland Rail Station. In order for a computer to make sense of this, a unique name is necessary to prevent it from advising the rail passenger from alighting from the bus.

4.3.3 Different coverage in services

Different areas of coverage for services exist. One type of coverage is a county-based service. A second type of service is based on mode or transport operator. The third is a city-based service. These are explained here but are further detailed in Chapter Five.

The county based service typically gives complete coverage for all trips within a county, and limited trip information for journeys to and from the county, and no information on trips between places outside the county, even though both places may physically be in the list of places in the county. It is crucial to emphasise that data within the county boundary may have a higher level of regularity and detail than data outside the county boundary but within the county's database. For instance, data within the county may include all 'timing points'. Outside the county, the database may only reference those stops that are 'principal points', for instance, major town (centres) for longer routes in rural areas. Even so, the record of the potential points where a person could alight and change to another service may be imprecise in addition to being labelled ambiguously and thus inaccurately. For instance, data of several counties do not include intermediate points as discussed at the beginning of Chapter Two, and that the TransXchange project sought to at least refine the issue. Consequently, it could arise that an adjacent county holds more precise data than the county whose data is being described.

National Express and Railtrack are examples of this type of service. Railtrack holds mostly rail data, but this does include bus services and ferry services which link parts of the rail network together (i.e. Ryde on the Isle of Wight with Portsmouth Harbour). Similarly, National Express includes some rail services, and also some coach services by other operators.

A third type of service provides information for just a particular city and works in a similar manner as the county service, but may have a different granularity or resolution. For instance, it might contain particular points of interest or have different naming conventions such as employing street names instead of relying on the traveller to know where (or rather, which of the many bus stops) is Soho (in London).

4.3.4 Different methods of data storage

Several methods of storing trip information and relevant details exist. These are protected by companies' Intellectual Property Rights (IPR), although it is sufficient to state that the differences are significant and often not resolvable except to convert data to a format named CIF, (see section 2.5) based on the standard spelt out by British Rail or ATCO (Ness, 1995-1999).§

§http://www.users.globalnet.co.uk/~jplanner/datag.html#cif
Certain methods of arranging data into structures have advantages over others. These advantages could be temporal and realised through faster data retrieval and thus response or faster calculation of a route. This advantage has market value and thus companies wish to protect their intellectual property rights.

This possible infringement of a particular company's IPR further encourages a common format of data exchange that is not proprietary to any one organisation.

4.3.5 Different variable names

Within the Hypertext Markup Language, a method of assigning values to variables exists. Several methods exist to do this. One method uses a radio button, allowing the user to assign the value corresponding to a particular button to a variable. A second method is a textbox allowing the user to key-in text. This text is then assigned to the variable corresponding to the textbox. A third method allows the user to select from a drop-down list. The user then assigns the selected list item to the value of the drop-down list object.

The commonality of these methods is that each of these methods assigns a value to a keyword. This results in a keyword=value pair. For instance, a textbox may have a name of ORIGIN. The user might key-in a value of SOUTHAMPTON. The resulting keyword=value pair would be "ORIGIN=SOUTHAMPTON". This formatting is consistent with HTML. However, the naming of the keyword is NOT consistent and thus requires a protocol. Several different methods of naming fields and their values exist. The table below shows some examples.

<table>
<thead>
<tr>
<th>Company/organisation</th>
<th>Origin</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pindar</td>
<td>From</td>
<td>End</td>
</tr>
<tr>
<td>Railtrack</td>
<td>Start</td>
<td>Destination</td>
</tr>
<tr>
<td>Romanse</td>
<td>CRITERIA</td>
<td>CRITERIA2</td>
</tr>
<tr>
<td>Hafas</td>
<td>SalitOrt</td>
<td>ZAltOa</td>
</tr>
<tr>
<td>national express</td>
<td>A</td>
<td>Z</td>
</tr>
</tbody>
</table>

*Table 12: Keywords used for 'origin' and 'destination'*

It is perfectly permissible for journey planners to use different keywords to mean the same thing in communicating with the user. However, should the journey planner wish to communicate with another journey planner directly, for the sake of cross-interrogation, it becomes obvious that this internal communication should follow some standard.

Another issue to contend with is differences in how the data within these fields are formatted. For instance, several different methods of representing dates, times, and other parameters exist. In particular, the convention in the United States presumes that the day and month come in the opposite order to the convention in Europe. Also, time is sometimes depicted on a 24 hour basis instead of the 12 hour plus AM or PM convention. Table 13 shows some examples.
### Table 13: Several Date and Time formats

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mm/dd/yy</td>
<td>hh:mm</td>
</tr>
<tr>
<td>Mm/dd/yyyy</td>
<td>hh-mm</td>
</tr>
<tr>
<td>Dd/mm/yy</td>
<td>Hh</td>
</tr>
<tr>
<td>Dd/mm/yyyy</td>
<td>Mm</td>
</tr>
</tbody>
</table>

As in the previous section, whilst it is permissible to allow unique date and time stamping between the user and the journey planner, a common method of standardisation is necessary in communication between journey planners.

#### 4.4 Message Format

ASN, EDIFACT, TRANSMODEL, and DATEX are possible candidates to form the basis for a message format within any transmitted data. However, available documentation for ASN suggests it was more concerned with road related operations. EDIFACT appeared to be aimed at the air and ferry markets, whilst TRANSMODEL aimed at the bus (operations) market. DATEX includes a small section on public transport which can be used for more than one mode, and the ATCO.CIF definitions also allowed transfer of data about more than one mode. It was still necessary to refine these into a separate, derivative protocol for public transport data exchange. Thus by taking the best aspects of DATEX and ATCO.CIF, it was possible to create a list of specific labels for data, and group them using a structure or hierarchy similar to that shown within DATEX and Address Point.

Whilst it would have been sufficient just to exchange the data in a flat string, ambiguity remained as to the exact interpretation of some the exchanged data. It is here that HTTP (see 4.1.5) has a feature which is particular useful. It has a Markup feature allowing parameters to describe values which remove the ambiguity of a particular set of data. It also includes the keywords in the transmission, facilitating the interpretation of the data by the second party without necessarily needing a copy of the 'data dictionary'.

It was thus possible to create a new “HTML” for public transport by suggesting that the format of the string being exchanged or passed between computers be comprised of by combined three protocols.

The first protocol is the TCP/IP message protocol. Next, the string follows the arguments of the HTTP/1.0 message protocol and the required methodology. Finally, the data string must follow the arguments of what is to become the 'JourneyWeb' protocol.

Whilst the TCP/IP and HTTP standards are well documented elsewhere, no precedent existed for using a modification of HTML to exchange data for public transport (later known as the JourneyWeb protocol). However, it was not an enormous leap of faith to hypothesis that should it be possible to define and create tags with different names than those allowed, embed them in HTTP style messages, and that this
innovation would be enormously useful. Thus the format of the JourneyWeb protocol mimics that of HTTP/1.0 as follows.

```xml
[element name]data[element name]
```

The format of an embedded element would look as follows, where "e1" is the name of the first element and "e2" is the name of the second, embedded element.

```xml
<e1>e2>data</e2>e2>data</e2></e1>
```

For example, HTML version 2.0 has a structure allowing the presentation of data in a table format. It is constructed via this hierarchy.

```xml
<table>
<tr>
<td>
```

In this manner, a table is defined. It may have multiple rows. A row may have multiple columns or fields. Specific rules exist for handling consecutive rows where the number of columns or fields may not be uniform.

By extrapolation, it was possible to suggest that a similar hierarchy might exist in transport data definitions, as discussed in Chapter Two. An example illustrates how this hierarchy was extended to public transport data, allowing the display of information easily within HTML. This hierarchy is as follows:

```xml
<response>
  <journey>
    <trip>
      <field>
```

Consequently, a "response" can have multiple "journey" solutions. Each journey solution may have multiple trips or individual links within that journey, and finally, each trip has certain definable characteristics, such as its origin, origin platform if there is one, destination, departure time, arrival time and so forth.

By using this particular methodology, the receiving journey planner software (sometimes called the public transport navigator) can then rearrange the information within its software by storing it to an internal "array" or a allocation of addresses within computer’s memory. The public transport navigator can then format the data from the array as it likes and present the journey details in the manner it best sees fit, whilst still maintaining data impartiality. For instance, it might adopt a particular style of presenting in HTML and send the information to a browser. If the recipient is a (mobile) phone, then the navigator might not bother with anything but text and a 24 character line across the screen.

Alternatively, a navigator can omit information, or use a series of screens to display the information depending upon its own preferences. However, the message sent across the wires in response to a request contains as much information as required at the level of data integrity agreed by the partnership.
<trip_definition>
  <origin>London</origin>
  <destination>Southampton</destination>
  <date>25/12/1995</date>
  <time>12:00</time>
  <deparr>Depart</deparr>
</trip_definition>

Figure 13: Sample XML showing service information

With respect to use for public transport, a short example (above) best illustrates the usage of this innovation.

Within this example, the standard trip definition can be exchanged between journey planner suppliers in an autonomous manner, using common names as “keyword” identifiers and normal, strictly formatted methods displaying the values between the keywords. So in this example, the origin would be London, with a destination in Southampton, on Christmas Day in 1995, departing after twelve noon.

It is salient to note the following constraints. Certain characters are not allowed: "=",”,”,”,”,” linear white space, and "&". These characters have special meanings under CGI. Furthermore, the data must contain the "+" character where one would expect a space when issuing a "GET" request. The "&" symbol joins keyword-value pairs together. The "=" character joins the keyword to its value (e.g. "keyword=value"). Finally, the "%" symbol is used to demarcate that the following two characters are in hexadecimal format.

It is now known that others thought of a similar version of this innovation independently from circa 1998 as eXtendend Markup Language (XML), with an initial publication on the WWW in November 1996 (Bray, 1996). However, they had not yet used it for personal information for routing via public transport.

4.5 Summary

Thus far, the research has determined that a distributed architecture had advantages over the warehouse approach. Thus, it was necessary to transport the data in a heterogeneous computer environment. Rather than use the seven layer ISO reference model, it was decided to use the Internet Protocol. TCP/IP was more flexible than UDP and of the TCP/IP methods of data messaging, it was decided to use HTTP as it had a number of features. Potentially, it could 1) allow concurrent requests, 2) provide information about any remote location through stacking requests, and 3) allow a common request from different types of devices. Rather than just send the data that might be ambiguous, a derivative of HTML was created specifically for public transport using keywords derived from DATEX and ATCO.CIF. In theory, this approach could then allow each traveller to concurrently request multiple types of heterogeneous data (i.e. schedule and fare) about heterogeneous modes of public transport (i.e. train and taxi) and receive an integrated response. However, to accomplish this, several message 'types' are necessary, as discussed in Chapter Two, section Four. Initially, three main types of messages are most important. Messages are
necessary to exchange data to answer the interrogation of a journey planner for data that relates places to one another. Messages are also necessary which relate data containing exchange points to origins or destinations. Finally, messages are necessary to define a journey definition and to return a journey solution. Building on the messaging structures discussed thus far, Chapters Five and Six examine the methods that might utilise these concepts in detail.
Chapter 5 – Review of Methodologies

5.1 Introduction

The purpose of this chapter is to develop a general methodology for journey planning, within a distributed, multiple mode, and heterogeneous data source environment. Besides works by McCormack and those he references, no additional research was found regarding "route" or "journey" planning with respect to public transport in academic literature which had been published by 1995. Since then, research by, but not limited to, Deutsche Bahn (Schnittger, 1999) and the Dutch Ministry of Transport (Potgraven, 1996) have become increasing available in English because of European funding through IST programmes. However, due to the recent availability of this information, the influence of McCormack's work on this research is more significant.

5.1.1 McCormack's methods

Documentation for McCormack's trip planner prototype includes four volumes, of which the documents 94.6 and 94.5 are the most relevant. He proposes an object-oriented, multiple mode trip planner system, which includes public transport legs and private transport legs. Section 4.11 of document 94.5 describes the methodology for route finding, but does not clearly specify if both unscheduled (private transport) links and scheduled (public transport) links can be combined in one journey.

The example given does quotes a long distance journey between one metro area and another, using unscheduled links (car, push-bike or walking). It suggests that if the metropolitan areas are "far apart", is it best to find the route between the two metropolitan areas first, and then attempt to look at the individual metro areas for local connections. These three links are then connected together for a composite itinerary. No methodology is evident for a public transport only journey, nor the specific types of journey which are discussed in this research. However, these may be included in the logic of the software code included in the document 94.6. Significant work was included in data input and a geographical interface, and this would complement work done here.

Whilst the theory of McCormack's trip planner is applicable to this research, several presumptions in place for his work are unacceptable here. For example, his documentation presumes a single data source, inclusive of a homogeneous data structure, and common data strings. This presumption is, whilst largely technical, a major obstacle of this research due to the large variety of data formats available. Some efforts at conversion have come about since this research started and this is discussed later. And whilst a single database for a small area that McCormack presumed might not have been an obstacle, a single gigantic schedule database at a national or global scale was an issue.

McCormack's work also presumes an object-oriented approach to software programming by all software suppliers. This research presumes numerous approaches to software programming, on many different
operating systems. Whilst this variety of operating systems is not an overriding constraint, they make the exchange of data cumbersome. Initiatives such as CORBA (http://www.corba.org/) are removing technical obstacles, but the level of co-operation could be perceived as inhibiting.

His research avoids another issue by adopting a common hierarchy for naming, and his hierarchy is directly correlated to the area covered by the data source. This presumption avoids a large number of obstacles. As such, a substantial portion of this chapter discusses just this issue. A related presumption is that all the points in the database have a geographic reference, and that this reference is of a common format throughout the database. Neither of these presumptions are valid here. Thus whilst it was possible to incorporate into this research study the basis theory of journey planning developed by McCormack, it was necessary to develop areas he avoided to accommodate the heterogeneous requirements of this research.

5.1.2 Industrial methods

Many proprietary trip planning software and algorithms exist. Examples include that used by Hacon in Germany (http://www.hafas.de/), and a prototype system in Australia called TransInfo in Brisbane (http://www.opcom.com/). Whilst some publications exist in relation to these companies and their products, access to them was not available at the time of this literature review, and obviously, it is not all in English.

Numerous companies within the British industry use a variety of methods to find journeys. Examples of these companies include WS Atkins, Pindar Routel, Software Logic, Anite, Teleride, Cap Gemini, Sema, MVA, Figurehead, Logan Interactive, FWT, and Southern Vectis. The methods behind these systems are proprietary and not published.

In general terms however, all surveyed journey planners define a journey first. Then the journey planner searches through its database of timetables to select and propose a solution according to pre-determined criteria that the traveller may choose. Some county councils such as Buckinghamshire, (Slevin, 1995-9) go a step further to try and aid the traveller by determining some criteria in advance, on behalf of the traveller, in the design of the software.

To determine the journey definition, the journey planner normally takes as its basis all “places” within its database. The traveller then matches against these using one or more methods as described in Chapter Two. The same is true for the date and time, and usually includes the parameter “to arrive before” or “to depart after” also. The journey planner searches for direct routes between these places as delimited by a “route”, and then uses the parameters of time to select a particular “service”. Should the journey planner not find a direct route, it must try to combine two or more routes via an exchange point, which usually occurs at a “cluster”. 
In the “early” days of this research (1996), querying a database to combine routes took “too long”. Thus interviews with two suppliers of proprietary journey planner software in 1996 revealed that it is sometimes necessary to compile lists of solutions. These lists can be indexed, and then later searches by the journey search engine could be achieved more quickly. Another supplier later suggested that a better method of arranging schedule data existed, warranting the others “inefficient”. Whether this is true or not is would be a matter for independent research.

Whilst this proprietary, “indexing” methodology was apparently commercially satisfactory for an area bounded by rural counties and local authorities in 1996, extending these processes beyond a particular area proved to be “undesirable” with respect to response time. On the other hand, hardware technology continues to advance at a significant pace even today (2000) making larger “areas” possible. This area then might extend to become a “region” or larger.

However, as the area gets larger, it is assumed that the number of “clusters” of viable exchange point increases. Presumably, the number of data indexes necessary for a pre-compiled solution using indexes also increases as a result of a large area. In “areas” with frequent services, where arguably no need for journey planning exists according to some transport operators (ATCO, 1999), the indexes may become quite large and cumbersome. Even if, for instance, the whole of the UK could be placed in a single database, the problem still exists for those travelling locally to nearby areas such as the north of France and Ireland. Thus this research returns to the premise that a distributed system must exist for reasons of extensibility, and indicators of the potential impact of this decision will be discussed later. Before sense can be made of these indicators, it is necessary to investigate just how one could plan a journey in a heterogeneous trip planning system.

5.2 Journey Planning

Following the precedent of industrial software applications and previous research (McCormack, 1994), it is suggested that two major functions are necessary to plan a journey: to define the journey definition, and to determine a selection of alternatives.

To enable this, some statements made at the end of Chapter Four and some of the avoided issues above are revisited. In this heterogeneous computer, data and network system, it is necessary to suggest some fundamental, common processes enabling these two functions to occur. Thus, it is suggested that at least three dialogues exist between a "speaker" and a "listener" within a conversation. Publications by Eurospin (Booth, H., 2000) and EU-Spirit (Schnitter, 1999) concur in principle, but it is known that their research was not independent of influence from the research described in this thesis.
These three dialogues are as follows:

• asking for and receiving further detail regarding the definition of a “place”
• asking for and receiving a selection of exchange points as necessary
• asking for and receiving a selection of journeys

In each of these dialogues, it is necessary to address several issues with respect to the data requirements. For instance, it is necessary to determine what data is needed and what data is readily available in a practical format. Issues with data arise from this investigation, as discussed in Chapter Two. Issues with the architecture and messaging were discussed in Chapters Three and Four.

However, a necessary feature set out in the objective was to enable a journey planner or system of journey planning components to have the potential ability to calculate a “cross boundary journey”. To review, existing public transport journey planners within the survey of current technology (Chapter Two) limited travellers to planning journeys with a specific boundary, typically a county or an urban area, or by a specific mode. However, in the fragmented, multiple supplier market of public transport information provision, the UK national government was interested in an integrated approach which could glue existing journey planners together, and yet allow new entrants enter in an incremental fashion, thereby filling remaining gaps.

At this point it is useful build on the Internet section in Chapter Four. As part of the Internet, computers find each other by one computer making a request to a “Domain Name Server” for a specific computer’s “address”. This allows a network of computers to find one another, in a similar manner as a letter carrier determines where to send the post from an address on an envelope of a normal letter. Similarly, this research proposed to add another dialogue in the conversation discussed above.

• select the appropriate (address of) data source(s) for further information

By adding this dialogue, the journey planner can then cope with another dimension to the system, answering the question where further detail might reside. It is not clear though what the central resource might be, where it is, what it contains, or how big it must be. Nor is it clear how it is to be managed. However, returning to the problem of the geographic naming with respect to the political structure, it is possible to suggest that an analogy might form here.

5.3 Issues in creating the Journey Definition

The aim of the Enquiry Engine is to assist the traveller in defining a journey definition. As stated before, this involves determining the origin, the destination, the date, the time, and whether the date and time is a limitation from which one is to arrive by this time or depart after this time. As there are fewer obstacles with defining date and time, this section discusses these obstacles first.
5.3.1 Obstacles in defining a date or time

Of the methods to refine the date and time, issues became evident 1) between the traveller and the software programmes that make up the system, 2) between the individual software programmes within the system, and 3) within individual software programmes. Of these, this research concentrates on issues relating to exchanging date and time data between software programmes within the system. However, it is relevant that the system must recognise that the traveller, when choosing a year, is currently picking a date in the future, and thus “00” would read 2000.

Between and within the software, it was necessary to be more precise in the definition of the date and time. Several reasons exist for this. Computers in different countries express dates and times differently. In the USA, the month precedes the day. In Europe, the day precedes the month. Different methods of internal date storage exist within software and this varies by manufacturer. Some start as the first day in 1900 as 1 and number days serially there after. So some date in November 1999 would read something like 36009. Yet another reason for differences is the concept of GMT, which will not be reiterated here.

Timetables however change frequently, so it has been practice to allow the user to choose only a date in the future, and never more than a few months in advance. Thus, systems tend only to allow the definition of a date for the next few weeks, or, in the case of limited timetable validity, to allow dates within the validity of the timetable.

With respect to time, most journeys occur between the hours of 6 AM and 22:00 (10:00 PM). As illustrated though, the representation can be as “military time” or the standard AM and PM. Internally to the system, times are sometimes understood as minutes past 0000 (midnight), without the semi-colon. So 8 AM might be 240 minutes past midnight. It could be argued once that this was for efficiency of searches. Conversion between string and numeric types of variables is extra work and string variables take more memory space. Whilst this may indeed be true, its relative significance is not felt to be important, and depending on the relative size of the measuring tool, it may be difficult to prove given the processing speed of current processors, memory and inexpensive yet fast storage devices.

To summarise, a number of issues exist with respect to dates and times. These issues are largely a matter of agreeing a common standard where necessary, and letting the market decide elsewhere, as technologically, the issues are minor. In practice this has meant adopting a common date and time standard, but for testing, a simple date format (dd/mm/yyyy) and time (hh:mm) were used. However, new systems should be capable of identifying common formats even if they are not the agreed standard and react accordingly.
5.3.2 Obstacles in defining an origin or destination

Perhaps the most practical obstacle to defining an origin or destination is that transport service operators, councils, and travellers all interpret a name of each place, building, station, bus stop, etc., in a slightly or even significantly different way to every one else. For instance, London is not a place that is readily defined. Transport for London (Bourne and Harris, 2000) suggests that their database has about 90,000 uniquely definable points, (e.g. bus stops, platforms, rail stations, airports, points of interest, and so forth). Inclusion of the street addresses would raise this significantly.

From a different perspective, it could be said that Greater London has as many different points as it has abodes and offices, in addition to the points of interest and transport network points. This collection of points could add several million more. Even (central) “London” from a rail perspective could be defined as one of more than 19 principal, main-line railway stations, (each a small city itself perhaps), or one of five or more principal coach stops or station. Some 16,000 Underground (tube/metro) stations and roadside bus stops exist (Bourne and Harris, 2000). This ambiguity has proved to be one of the biggest obstacles in the implementation of an extensible system.

**Figure 14: Translation of Operator Names to Common Name**

An example illustrates this point more clearly. A journey planner dealing exclusively with the rail network in Britain may refer to “London” as meaning the 19 or so main stations in Central London. Thus a journey from “Watford” (rail station) to a rail station in (central) “London” would implicitly suggest that it is in fact journey between the two rail stations. A similar journey definition would also fit the journey planner for London Transport (the tube). A journey planner serving a county council however would be expected to have multiple modes, as it might suggest that a traveller would desire to go between Bristol and Bath for instance.

As it happens, a traveller has a great deal of choice in travelling between the (town centre of) Bristol to (the town centre of) Bath. He or she can cycle, catch a non-stop train, catch a “fast” bus, take the coach,
meander via a scenic bus journey, or even walk or take a cab. At some times in the year, journey by canal is even possible. A combination of these modes is also possible, and may even be required, as the station in Bristol is not directly in the city centre. The journey planner may only provide the option of choosing “Bath” to “Bristol”, leaving the traveller to guess what is meant by these quite large and ambiguous names. Figure 13 illustrates the case where a series of similar text strings might be near enough for a human to guess the common name, but a computer would have to be told that they are in fact the same place. To be even more precise, these names may have to refer to the point at the southern, passenger entrance of the station (the station has several entrances, one for post, and two for passengers, a service entrance and an entrance for those that work in the offices above the station).

It is increasingly obvious that some form of naming hierarchy is necessary. If every “name” of a place, and its local variations and spellings, in addition to all the addresses and points of interest were included, it has been determined that at least 30 million “names” might exist. Ordnance Survey and the Royal Mail suggest that 26,000,000 addresses exist even without points of interest. This suggests that it is necessary to look again at the political structure to see if it is possible to mimic the hierarchies suggested therein in a meaningful manner. At an as yet undefined level, it is suggested that these “names” at the appropriate level be apportioned an address or reference to further “detail” or names in subsequent levels.

Thus, using the analogy of the DNS from the Internet above (in the last paragraphs of the previous section 5.2), it was suggested that this reference take the form of an “Internet!” address, which can link or “point” to a list of “places” within the name at the higher level. For instance, regions have administrative areas in them, and administrative areas such as county councils and local authorities, have communities or villages “within” them. It is not absolute however, as the area of a given local village may straddle two or more higher level administrative areas. Thus assuming that it is appropriate to allocate references at the regional level, a small relationship table can be created relating the region to an Internet address. The Internet address provides the reference or "hook" that allows one to link to lists of further administrative areas in this scenario.

5.3.3 Maintaining a record of past user-decisions

Some method of maintaining the integrity of the travellers’ choices is necessary. As discussed earlier, the research described in this thesis used an independently derived subset of HTTP (see Chapter Four). Another method of interoperability exists (i.e. CORBA, mentioned in 2.4.10—see "object request broker") though. However, this object-oriented approach to interoperability is slightly more complex in its approach because of the manner in which it maintains the “state” of current processing events. At the time this research was undertaken, the resources required to examine the difficulties that the implementers would experience should CORBA be adopted to overcome issues of heterogeneity in operating systems and data types did not yet exist. Regardless, it remains true that the computer competence increases significantly, whilst the “readability” of the protocol decreases. Therefore, the issue of integrity is discussed with relation to this method (XML) of implementing interoperability.
In HTTP, one method of maintaining the integrity of the previous decision is to use the “navigation” of links to create a “web” of text files with references to subsequent text files. This referencing would result in hundreds or thousands of individual text files, and a large matrix of navigation paths. Presume, for instance, that a text page published on a “website” has a list of regions, and each region is then linked to a list of inclusive administrative areas. So if the traveller choose the “branch” of the navigation labelled the Southeast, this would link to one of ten pages (assuming ten regions) with a list of administrative areas. Assuming nineteen administrative areas in the Southeast, leads to a further nineteen text pages, further down the “web”. Choosing a particular administrative area would lead to a further list of villages, and if there were say 500 villages in a county, then there is the need for 500 more text pages.

Thus far the “web” has one plus ten plus nineteen plus 500 or 530 text files, not including the other “branches” of the navigation. Now the sample text page in Figure 14 has 7,724 bytes. Thus a branch of 530 files equals about four megabytes of data.

Now a traveller has to choose a destination in addition to the origin. Each of the previous villages would have to then start a new “tree” to choose the destination, with each village starting the same new tree, and thus maintaining the “memory” of the travellers’ individual choice. This method is not practical because of the large quantity of storage necessary to achieve the aim. Besides the cost of the physical data storage hardware, maintenance of this method could prove tedious.

A second method is to use a “form”, which contains imbedded information, and a “script”. The form is a preformatted text file of HTML with text boxes for inputting data or drop-down lists of options, from which the user can choose a suitable record. After selection, the “form” is submitted or “posted” much like one posts a letter in a letterbox. Text fields (see below with reference to “cookie” and “hidden text box”) on the form allow the software system developer to store values from previous actions. The script acts as a fancy letterbox, receiving the “form” and its information, decodes it, processes it, and takes the appropriate action, perhaps returning a letter in an HTML envelope to the user via the display within the browser software program.

Using a second feature of scripting, the “letterbox” is able to interact with other data resources such as a local database. In the scenario of choosing an origin, the letterbox a.k.a. script is able to see that the Southeast region is the relevant parameter value. Using the address on the form of the envelope, the
letterbox queries a database at that address (it need not be local) to determine what administrative areas lie within the Southeast region. The database returns the list and the letterbox appends the necessary information to the form, keeping track of what has happened thus far in this dialogue. This process can thus continue without hundreds or thousands of individual text files necessary to map out the alternative scenario. It is worth repeating that this option is only viable in a “stateless” interoperable system such as discussed here if a method of maintaining data integrity exists.

At least three methods exist to maintain the memory of the previous action. One method is to use a “cookie”, which is a file that resides on the client’s machine. The second is to embed the information in the form using what are called “hidden” text fields. The third is to maintain a log on the server of all transactions and their values and query that each time.

The advantage of a cookie from the perspective of the server administrator is that it saves space (£) on the server data storage system. From the client’s perspective, a file is written to the local machine that must be deleted at some stage to save space locally. The advantage of the hidden field is that it does not require a file to be written anywhere. A drawback of this hidden field is that if the information is confidential, it is easy to see the value within the hidden field. The advantage of keeping a log file is that the administrator of the server knows the state of every dialogue as it progresses all the time. The disadvantage of this is that it adds to data maintenance and again costs money in data storage space.

This thesis has discussed the technical consequences of navigating through a system of origin and destination options using either a system of files or a form, script, and database method, and maintaining a record of what decisions had been made. Whilst it is possible to use the former method, practical issues such as cost of data storage hardware suggest that it might be advisable to use the latter method. Whilst the software and logic to do this are more complicated, it could mean a substantial savings in hardware procurement by any one or more organisations. Even given the difficulties in linking the database to the software and subsequent complexities in writing the software, scripting is now the accepted method of practice.

5.3.4 Mitigation of Issues in Name Definition

Having taken the decision on how to store the data, this thesis returns to a primary issue: several names can exist for the same point; and one name can reference several points. Thus, it was necessary to develop an extensible method to uniquely identify the physical location of a point. This research thus developed three methods for consideration.

- continent.country.region.state.county.city.village.cluster.bay.point
- altitude.latitude.longitude
- continent.country.region.state.county.city.village.address.postcode
Whilst these methods may be highly useful, they are also idealistic. Current databases support simpler forms of the first method. Further, trials suggest that the first and last may in fact be combined in order to accommodate large transport terminals such as airports and ferry terminals. This results in the proposal of,

\[
\text{continent.country.region.state.county.city.village,}
\]

and

\[
\text{street.crossroad.object/building.terminal.wing.gate,}
\]

where terminal, wing and gate and may be exchanged for cluster and bay.

A further practical problem arises with this structure. A database table with all points within Britain would result in reference to over 30,000,000 places, including addresses, points of interest, stops between timing points and normal timing points are also included. Therefore some method of breaking up the database of tables of places is necessary.

A hierarchical naming or “gazetteering” system is thus proposed, whereby, starting at the top entry, Earth, an enquiry can be made to a standard table as to the location of remote data services holding data on the next level down; i.e. Europe, Asia, and so forth. At each level then, a list of remote data services is related to each country in a list of countries for each continent in a one to many relationship. Therefore an address of further detail within the name is associated to each name. This process is then repeated until one arrives at the City or Village level.

At this level, more detailed information is necessary, including where to find a list of all locations or points within a given area. The concept of a gazetteer was introduced. The gazetteer is an index relating ‘villages’ of localities to where additional information may be found, such as places in that village. One may then request and retrieve detailed information from this location (i.e. an internet script address).

Another practical issue arose as numerous points exist in rural areas within a region or county where a point may be directly related to a county or region, and not necessarily with a village or city. However, in England all points fall within a specified administrative area. Further, this administrative area, usually a county council, has responsibility for all “wards”, precincts, parishes, villas, hamlets and so forth within its boundaries. It is then generally possible to assign a data supplier to each administrative area, and the “place” to the appropriate administrative area.

So for instance, Amerham, Aylesbury, Chesham, High Wycombe and Wycombe might all be associated with the county of Buckinghamshire. The county of Buckinghamshire might then be related to this URL: “http://www.soton.ac.uk/~pti/bucks/pjp.cgi”. Then by JOINing these two relationships within a database, it allows the navigational journey planner to allow a relationship between the “place” and the address of the data supplier(s’) software program. The navigational journey planner might then request further information such as places within Amersham (just a few), its exchange points (perhaps five or ten) and any potential journeys.
5.3.5 Summary

Several obstacles and issues were discussed in this section, and after some discussion of existing practise (based largely on interviews with those in the industry), some tentative solutions were suggested. One of the concepts discussed was that of the gazetteer. The purpose of the gazetteer is to assist the journey planner to uniquely define a place by setting up a series of links to further information, such as places for villages, exchange points, and journeys from search engines. It is now possible to collect all the necessary information to determine a journey definition, and where to go for information which might provide some information towards a solution. However, it is not yet clear how an itinerary or a selection of journey solutions might be determined.

5.4 Obstacles in determining an Itinerary

Several issues were encountered relating to allowing the journey planner to make remote Internet requests based on the HTTP derivative of TCP/IP. It is repeated here that no previous research on the subject had been found apart from McCormack, although similar products exist in Australia and Germany. It is helpful though to acknowledge that the design and development of this section benefited from discussions with the suppliers of data systems in the industry. Of particular importance were discussions with the rail infrastructure company at the time, the national coach supplier, the data suppliers for the councils, and members of the councils themselves.

By way of introduction, the different types of journeys are discussed. Then the heterogeneity of available data is described, and subsequently the consequences for journey planning are considered. Thereafter several theoretical methods of composing requests are discussed, preceding the final section proposing a generic methodology.

5.4.1 Journey Types and Routeing

To achieve anticipated aims of the government, this thesis sought to design a “system” which would allow a traveller to obtain a journey itinerary between any two places. Logically, this could mean a “local” journey, a “mid distance” journey, and a “national” journey. “International” journeys are also possible.

*The crucial aspects of paramount importance are not these ambiguous labels of “local” or “international” in a political sense, but the boundary of the database area in a physical sense. For instance, a route may start in one political area and end in another political area, as in Figure 15.*
If both area A and area B are in the same database area, then a journey between a point on the 66 service in area A to another point on the 66 service in area B could be labelled as a "local" journey from the perspective of the database area. However, geographically speaking, these distances could be quite "large", spanning 40 or more kilometres, and thus be classified as mid-distance or long distance journey from the perspective of the traveller.

It is the case that if area A and area B are in different database areas that has caused many long-standing debates. If a service effectively links these two database areas, then effectively one has the possibility of a journey joining these database areas, even though the political areas are not adjacent to one another. In Figure 16, service 76 illustrates this between areas A and C.

If these two database areas are not adjoining, then the service joins two non-adjacent areas both in the political sense, and from the perspective of the database areas. For instance, if an area D existed to the right of area C, and was in database area 3, then a potential service could link area A and area D in two non-adjacent databases. An aeroplane journey is a prime example of this, if one limits the journey planning to one particular mode.

In summary, it is useful to re-examine these classifications as "local" to one database area, "mid-distance" as between two "adjacent" or adjoining database areas, and national as between two non-adjointing databases. This presents a challenge however, as the journey definition and so called gazetteer define the ambiguous term of a "place" using political boundaries and not database boundaries. Whilst significant overlap between the database and the political boundaries exists to encourage temporary solutions which ignore the fundamental problem cited here, it remains a challenge to resolve this issue in a comprehensive manner.
5.4.2 Data Precision

Another issue that challenges this research derives itself again from the ambiguity of naming a stop, but more to the point, the issue arises from data within a database area but outside the boundary of a political area. To understand the concept fully, it is necessary to illustrate the difference between several different types of stops or places.

A service is composed of stops, which are sometimes ambiguously named as a "place". For this particular section, this thesis defines three types of stops on bus, coach or rail services. These are major stops, minor stops, and intermediate stops. A major stop is sometimes known as a timing point, that is, a point which a time for the arrival of the transport vehicle. This time is fixed in law, and the transport service operator is legally bound to have a transport vehicle at that "point" at that time. To allow for impediments such as traffic accidents, congestion, and anything else, the Traffic Commissioner allows five minutes either side without penalty.

However, if the point is named as The High Street, Winchester, the transport service operator has a great deal of latitude. This is because several physical bus stop poles might exist on the High Street. Thus the transport service operator can argue that the time associated in the timetable or on the registration with the High Street represents whichever of the poles that would allow the operator to avoid the penalty at the moment of inspection. Therefore, imprecision is encouraged.

Minor points can be classified as points necessary to allow the traveller to know what the time will be at a particular stop. These points are usually a few poles apart, as poles might only be a few minutes apart in an urban area. Intermediate stops are the time for each and every pole. Thus if all intermediate points were included, then the precision would be very high indeed. In the days of paper based systems, it was prohibitively expense to include all poles in the timetable.

The consequence of different levels of precision can be seen in the following example. Assume that a political area has a database, and that some services extend beyond the political boundary. It has been practice for the owner of the database area to provide detailed data regarding a point within the political boundary, and less detailed data outside the boundary (ATCO, 1998-2000)
For instance, if one takes the example of Shropshire, in the Midwest of England. To the east lies Staffordshire. In practice, the database area for Shropshire will contain some services that exist partly in Staffordshire. However, as Shropshire is paying for the database, detail of the service within Shropshire may include every major stop (timing points) and many minor stops. In urban areas, intermediate stops might also be included. Once the service crosses the boundary, extracting the detail for the service beyond the boundary requires communication with the neighbouring authority. They may speak a foreign language (Wales is to the west of Shropshire), or they may be some distance away if the county is large. As such, the detail of the service on the other side of the boundary, for a variety of reasons, may only contain the end point of the service, and might be as ambiguously labelled as "Birmingham". It may include some major stops (timing points) along the way, but not necessarily in any consistent manner.

The conclusion of these issues is that service details vary both within database areas and within political areas. The result of these numerous, consistently inconsistent data is that the fundamental building blocks of journey planning, i.e. the service details, are largely unusable from the perspective of the traveller should it be necessary to cross any sort of boundary. The implications for this research are that it is only possible to demonstrate journeys that meet particular requirements. However, this issue of data precision and consistency is being addressed between local authorities, between data suppliers, by the DETR (i.e. TransXchange, 2000), and the Traffic Commissioners.

5.4.3 Remote Journeys

In a distributed journey planner navigation system, some requests for journeys will be considered 'remote' in that the data necessary to respond to the enquiry will be elsewhere. It is necessary also to consider the case for a local journey that is remote from itself. For instance, presume the journey planner is in Cardiff, and the traveller, accessing it from anywhere, wishes to travel between two places in Central London.
The Cardiff-based planner needs to access the journey planner(s) responsible for London. It may need exchange points also.

For journeys which cross multiple boundaries, a journey planner needs to make at least several requests to remote data services to obtain a selection of individual trips or legs for each segment or stage of a multiple-leg journey. For instance, travelling from a place in Winchester to a place in Edinburgh would be considered a cross boundary journey. Presuming that a journey planner can request from itself the local schedule data for the first leg of the journey, i.e. to get from the local place in question to the local "national exchange point" for Winchester, the journey planner needs to make at least two remote requests:

- one request for journeys from the national exchange point in Winchester to the national exchange point for Edinburgh; and
- one request for journeys from the national exchange point in Edinburgh to the local place in Edinburgh

If both ends of the requested journey are effectively "remote" to itself, then the journey planner would need to make three remote requests for journeys. Only then could it splice together a selection of legs to form a comprehensive itinerary.

The critical point raised from this obstacle is that one application process will have to act as a 'broker' or 'host' for the traveller's request. This application process then talks to other application processes on remote computer systems to obtain enough information to enable it to respond to the traveller.

5.4.4 Exchange Points between Transport Networks

The previous section raised the issue that it may be necessary to link individual journey segments together. This need arises both within a particular mode of transport, or when one mode of transport is used to link two local segments together, i.e. a flight between a local bus and coach journey. Also, in a journey from a point in Wales to another in Scotland, a journey planner needs the ability to recognise that these two places are not in the same county, not in adjacent counties, but within the UK. Thus some method is necessary to enable a journey planner to link these exchange points, and then link up individual journey components (from remote systems) to present an integrated a journey solution.

As one possibility, if the definition of the two selected origin and destination places contain a reference to administrative information, a journey planner might determine that it needs a "national" journey segment to connect two places via appropriate national exchange points. For instance, if the origin is related to "Wales" and the destination is related to "South-West England", then the journey planner could look for databases which link Wales and South-West England. Alternatively, a journey planner might have access to geographical information and be able to compute an approximate distance between two points. It
would still need to access databases that might link the two points. If more than one possible set of exchange points exists, then a journey planner needs information about journeys between each pair of long-haul exchange points. Consequently, it is suggested that some sort of index allowing a journey planner to categorise a journey is necessary, in addition to allowing it to request possible exchange points.

5.4.5 Methodology for POST Requests

Issues of speed and capacity have led to questions about whether to send remote requests for journey solutions one at a time or as a matrix of multiple requests. In taking the decision to allow a matrix of requests, another limitation soon followed. In HTTP, it is again possible to use either the GET method or the POST method to send the request. With some Web server software, the length of the CGI_QueryString variable is limited under some circumstances to 255 characters. Even simple matrix examples exceed 100 characters. Therefore, it is necessary to use the POST method for remote requests for journey solutions.

Even given the inherent flexibility provided by using the POST request, two conflicting issues arose. The first issue was to provide the data requester with the greatest flexibility in composing the journey request. This issue conflicts with an earlier limitation of trying to limit the request size so as to minimise the Internet data transfer time. However, with the price of data transfer decreasing and the speed of data transfer increasing respectively with time (BT, 1998-9), it was considered to be better, within reason, to provide flexibility in the protocol rather than base the design of the system on the current cost and speed limitations of data transfer.

Several methods exist to compose and structure a request string. Given a matrix of journey requests from origins 1 to N, and to destinations 1 to N, at times 1 to T, and so forth, the string composition could be composed in at least three ways.

- It could be a series of annotated strings. For instance, the string could read, origin (O) 1 to destination (D) 1 at time (T), date (Dt), and so forth (DA), followed by origin 1 to destination 2 at time, date, and so forth.

<table>
<thead>
<tr>
<th>O1</th>
<th>D1</th>
<th>T1</th>
<th>Dt1</th>
<th>DA1</th>
</tr>
</thead>
<tbody>
<tr>
<td>O1</td>
<td>D2</td>
<td>T1</td>
<td>Dt1</td>
<td>DA1</td>
</tr>
<tr>
<td>O1</td>
<td>D3</td>
<td>T1</td>
<td>Dt1</td>
<td>DA1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

- It could be an abbreviated series of annotated strings

<table>
<thead>
<tr>
<th>O1</th>
<th>D1D2D3...</th>
<th>T1</th>
<th>Dt1</th>
<th>DA1</th>
</tr>
</thead>
<tbody>
<tr>
<td>O2</td>
<td>D1D2D3...</td>
<td>T1</td>
<td>Dt1</td>
<td>DA1</td>
</tr>
<tr>
<td>O3</td>
<td>D1D2D3...</td>
<td>T1</td>
<td>Dt1</td>
<td>DA1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
5.4.6 Routeing Field Structure within POST method

Several methods exist to structure the fields within the methodologies described above. One method is to use the key=value methodology as defined commonly in normal GET and POST statements. For instance, the composed text string might appear as origin1=value&origin2=value& and so forth. A second method is to use Extended Markup Language (XML) in composing the field. In this methodology, the structure of the field could look like <origin1>value</origin1> and so forth.

However, a third method to structure the field could combine these two methods. An example best illustrates this structure.

```
journey1=<origin>value</origin><destination>value</destination>...&
journey2=<origin>value</origin><destination>value</destination>...&
journey3=<origin>value</origin><destination>value</destination>...&
journey4=<origin>value</origin><destination>value</destination>...</deparr>
```

This structure allows the data service to individually parse each journey for the remote journey routeing. However, even this structure is too ambiguous, and Chapter 6 revisits this issue in further detail. It is to be noted here that XML evolved independently, and is thus slightly different. In XML, the request does not contain the "keyword=value" format developed here. It simply examines the "header" information for the string "text/xml" and if it contains this string, the application process begins parsing the XML that follows in the "message". Also XML allows only ONE top level keyword, where as the research for this thesis utilised several.

5.4.7 Summary

This section discussed a number of issues with respect to composing a composite itinerary from a number of databases or political areas, as well as issues with message composition and the methodology. It stated that different types of journeys exist but that the salient point was not so much what political area an origin and destination lay, but whether they shared the same database area. However, as was said before in the previous section on the journey definition, it is not the database area that the traveller relates to, but instead it is the political area. Therefore it is repeated once again the importance of a relationship between the political areas and the database areas, implemented in this research via the gazetteer. In this
way it may be possible both to achieve journeys which cross boundaries, but also encompass more than one mode (i.e. coach and bus, or coach and rail, or bus and air, taxi and ferry, etc).

5.5 Generic Approach

It has thus far been determined what functions a journey planner has, what data requirements and availability exist, an architecture, and some methods for transmitting data. The purpose of this section is sketch a methodology for using previously discussed concepts for journey planning. Afterward tests demonstrate the concepts with a series of apparatuses. Using an incremental approach, concepts are added together to demonstrate an extensible journey planning system that might overcome the limitations of crossing geographic boundaries, having more than one data supplier, and using multiple modes.

A basic methodology for simple "long distance" or non-adjacent database area journeys existed in some suppliers' software. Their systems typically allowed multiple database enquiry systems by matching local names to points on the Rail and Coach networks (Houghton, 1995-6). To add to existing technology, it was necessary to generalise this methodology and add impartiality to the algorithm.

It is stated at the onset that this methodology is continually being refined and improved with further research already in progress (RAPID, 1999-2002), so as to include more difficult scenarios than those discussed here. However, this methodology set up a useful foundation from which to expand understanding, thus meeting the extensibility requirements of the research described here.\(^\text{10}\)

5.5.1 Defining the Journey

In the sample of software applications surveyed in this research, journey planners for public transport normally start by presenting a standard welcoming banner within a “form” to the traveller. Included in this form are several “text boxes” or input fields where the traveller may enter any number of characters, or select from a predetermined list of options. As a general preference, the traveller was presented an input field for the origin and destination, and a list of options for the date, and time, and any other travel preferences. The bottom of the form then had a “submit” command button. Clicking the button or pressing “ENTER” or “RETURN” then activated the “home” journey planner. This home journey planner then interacts with the rest of the national system on behalf of the traveller.

In order for a national system of public journey planners to work, each journey planner must have access to a national set of place names from which the traveller can choose an origin and destination. It was decided at an early stage in the research that producing a list of all possible bus stops, in the country, would be unwieldy and require an immense amount of upkeep.

\(^{10}\) It is appropriate to note that previous publication of this particular section in conference articles had significant input from Dr. Anthony Lock so as translate 'concepts' into concise 'English'.

90
A hierarchical system has therefore been proposed which is based on a list of approximately 58,000 communities (villages, towns and districts of cities) obtained from the Office of National Statistics; from this list a 'National Gazetteer' was to be developed.\(^\text{11}\) The National Gazetteer is a relational database, linking each community to the Internet address of the journey planner relevant to the local authority with responsibility for travel in that area.

Using these processes in the prototype system, the home journey planner matched the input fields for the origin and the destination against a common list of 58,000 communities as listed in an index obtained from the Office of National Statistics. Any matches it found were then fed back to the traveller with the selected date and time. The traveller then has the option of choosing from these origin and destination lists of communities, and submits his choices to the home journey planner.

The home journey planner then does a "look-up" to find which county or unitary authority the respective community resides in the National Gazetteer. The home journey planner then contacts the remote journey planner associated with that county or unitary authority using the associated Internet address, and requests a selection of individual stops from that remote journey planner using a standard JourneyWeb Internet call named "PLACE". The remote journey planner responds to the "home" journey planner with a list of stops by using a method of data transcription that is based on eXtended Markup Language. The home journey planner then feeds this list back to the traveller.

5.5.2 Finding Solutions to the Journey

Once this iterative process of defining a journey is complete then, interaction with the traveller ceases until the traveller receives the solution from the home journey planner. However, the system still must complete a series of intermediate processes, including how to break up the journey definition into the relative components, and, where the system may locate the necessary solution to each component.

The first task, breaking up the journey definition has as defined in this project, three cases. One case is where the origin and destination both lay within the same database area. This could be a single mode journey such as coach only, or it could be a multiple mode journey within a small geographic area. It need not be anywhere in particular as long as it is the same database area.

The second case was where the origin lay in one area, and the destination lay in a directly linked area, either politically or via the database boundary. The third case was where it was necessary to link two areas using a national carrier. For a national journey, this would normally involve three stages:

1) A local bus trip from the origin to a mainline railway station, or long distance coach station

\(^{11}\) The database was developed by combining the data sets from the English and Welsh branch of the ONS with the Scottish branch, and adding various indices and supplementary data in the form of Internet Addresses of the data sources.
2) A long distance rail or coach journey

3) A local bus trip from a mainline railway station, or long distance coach station to the destination

This research took the approach that it needed to resolve the community name. The journey definition has within it a community name and a stop name, both of which are unique. From the community name, it was possible to determine which remote journey planner was relevant from the relationship between community names and journey planner areas, explained in greater detail in Chapter Six.

If both communities have a matching "area" within the national gazetteer, the home journey planner (its location need not relate to the geographic positions of the traveller, the origin or the destination) need only make a request to that system (it could be itself) for journeys between the origin and the destination. This scenario exists for journeys local to a database (where ever the database is) and single mode journeys.

If the two areas do not match, further questions need to be asked. A journey planner has information about particular stops in a community outside the political boundary of a particular database. If it is the case that the political and database areas coincide, the address for these areas are listed as a primary source. If it is the case that the political area is not the same as the database area, the address for database area is associated with every political area under its influence as a secondary data source in the "National Gazetteer". Via this relationship, it is expected to link a local journey to a direct, cross-boundary service. Methods to prototype these "two-stage, cross-boundary" journeys are discussed later in greater detail.

If a secondary journey planner is not allocated (i.e. a journey planner is not primary or secondary to the origin or destination), it is necessary to link the two local journeys via a national link. This journey is thus sometimes referred to as a "three or more stage" journey. In this case, each participating journey planner system keeps a suitable list of national exchange points, where the traveller can change from the local to the long distance modes of travel. This database is known as the local registry of exchange points. The home journey planner can now request a list of these national exchange points from the remote journey planners in a similar manner as it did to request places.

In either the two stage or three stage case, the "home" journey planner now has a set of points it can link together using journey details obtained from a remote journey planner, via the Internet address from the National Gazetteer. Using a defined standard request then, it formats its request for each journey leg to appropriate remote journey planner. Then it makes an Internet request using the third standard call, later named "SEARCH", to available journey planners. The origin to the exchange points are done first, followed by searches for journeys between sets of exchange points, and finally, from the far-end exchange points to the destination, as appropriate.

Once the home journey planner has received responses (via XML) from each of the remote journey planners, it may then begin to splice the journeys as it sees best. The final process of the home journey
planner is to translate the spliced journeys into a response for the traveller, using HTML. Formatting of this response may include highlighting various parts of the information, placing boxes around and or grouping data as appropriate, and so forth. The response is then fed back to the traveller. In earlier chapters, these processes were originally under the heading of the “presentation engine”.

It is important to note that, as the industry and councils further refine their data, it will be possible to iterate this process to expand the selection of origins and destination to include all intermediate stops. With further resources, it would be possible to link in addresses, buildings, landmarks, and maps easily (from a technical perspective, but still complicated from the perspective of data conformity).

5.5.3 Testing

The focus of the research described in this thesis has been to create a system of interrelated software and hardware components which could enable a solution to a "cross boundary" journey, as a first step in many. As discussed earlier in this chapter, even the term "cross boundary" is ambiguous as journeys exist which cross political boundaries, database boundaries, or journeys which do both. Given the sample data available, the research described here is limited to demonstrating a journey between two non-adjacent database areas, via a third database area of a second database supplier.

In the case of two non-adjoining databases, “local” journeys, by bus or local rail, were combined with “national” journeys by long distance rail or coach. Research described here also investigated the possibility of a collection of journey planners and exchange points which might enable it to prototype this cross-boundary, and indeed, multiple mode journey using the following scheme:

- local journeys from an origin to a set of local exchange points on the national network,
- from these local exchange points to a second set of local exchange points, and
- from the second set of exchange points to a final destination.

![Figure 19: three stage "cross boundary" journey – one software supplier, two modes](image)

It was decided that it would be useful to test this cross-boundary journey scheme using a collection of three journey planners from one supplier, before tackling the multiple supplier feature as a separate case. This single supplier scenario is described as the homogenous case, with two local journeys connected via a national journey. However, it is important to clarify that the local bus journey could be a local train or metro journey if appropriate, or one that combines a local bus trip with a local train trip. As per the definitions, specific to this thesis, a trip is movement on a particular bus or train service and a journey is a collection of one or more trips necessary to get from an origin to a given destination.
Another salient feature of the research described here was to examine the possibility of enabling one journey planner from one supplier to interrogate a journey planner from another supplier, and vice versa. This feature might be better described as allowing a public transport journey navigator to broker a selection of journey alternatives by splicing together a collection of trip components from journey planners of different modes and different suppliers. EU-Spirit (Schnittger, 2000) describes this feature as "interconnection". This feature was particularly important to allow the traveller to see an integrated answer, regardless of who supplied the journey details, or what technologies were employed. This scenario is thus described as the heterogeneous case, as it contains multiple suppliers of journey planners, and multiple modes of transport.

![Diagram of a three leg "cross boundary" network - multiple software suppliers, modes, and exchange points](image)

This figure illustrates a local journey between an origin and three exchange points. As before, these local journeys can have one or more "trips". The middle portion of the figure illustrates the journeys between the exchange points of the national carriers on the rail and coach network. Here, five journeys are requested: four of them via the rail network and one via the coach network. Any of these can have one or more trips. Were the coach network to have two exchange points at both ends, it would resemble the rail network matrix. Finally, the right side of the figure as above depicts the local journey at the other end, connecting the national exchange points to the final destination point, again via one or more local trips.

### 5.6 Summary

Four topics of digital conversation were defined in this chapter. Review of several relevant works suggests keywords that could be exchanged, primarily for the origin, destination, date, time and service characteristics. Further examinations discussed the functions of journey planners, its processes, characteristics necessary and methods of interoperability (XML), and set out some basic definitions that are detailed in the following chapter. Various issues are then discussed, including specific issues and fundamentals of the different types of journeys and methods of journey routeing.

*Of paramount importance is the need to distinguish between the database area and the political area for a "place" along a particular service, and its unique identification being used by BOTH databases.*

Finally, analysis of these obstacles established some guidelines leading to the proposal of a generic methodology from which to prototype a working system, in which some aspects of the design will be tested.
With limited resources, the research described here is limited in what possibilities it might attempt to test rigorously. Without comprehensive data, it is further constrained to providing an indication of how a system with this functionality could behave whilst still meeting all the criteria, as illustrated in the figure above. Integral to providing an indication of this behaviour is the need to examine the fundamental characteristics of the design decisions (architecture, messaging) discussed thus far and their relative importance, given an indication of the merits in implementing the system one way or another.
6.1 Introduction

To overcome limitations of administrative boundaries and multiple suppliers, three areas of literature review were examined. In the first, a pool of data definitions resulted from an evaluation of existing applications, and research that occurred in parallel (i.e. Eurospin, EU-Spirit). The second review investigated the available data philosophies and structures, concluding on a particular structure for testing. Chapter Four examined the various messaging protocols, and modified these in an innovative manner to suit the requirements of these objectives. In the previous chapter, the methodologies and issues with respect to the necessary processes allowing (a collection of) computers to evaluate and determine one or more journey itineraries for a traveller, given a set definition, were reviewed and discussed.

This chapter describes the computer hardware and bespoke software systems (hereafter just "system") and tests, which hope to overcome the obstacles set out in the objectives in chapter one. It begins with a simple system, which prototyped the initial concepts of the Internet in 1995. Further systems were constructed and are detailed here, looking at bridging the gap between various journey planner components. Finally, a prototype system is described which aimed at demonstrating all the processes together. Future research might then look at modifications to this prototype to determine whether it can be usefully applied in a real-world, commercial environment.

6.1.1 Internet and Web System

This section examines a series of software applications and set up several methodologies to test if a data string could be sent from one software or operating system to another, regardless of manufacturer. This feature is crucial if the research is to overcome the restriction of allowing a journey planner to access information from more than one data supplier's database system. Initial reviews included Echo, Email, FTP, and WWW. All methodologies showed a "quick" response, that is, almost immediate for small text strings (about 5 kilobytes or less).

It was then necessary to set up and test each method of data exchange protocol. Due to limited resources, only two of the four options were set up in detail: Echo and WWW. The Echo server was set up as a learning tool as it was relatively straightforward, and free software was readily available, if only by chance, to test it. It also had similar characteristics to Email, and enabled a basis for more complex and expensive WWW. For ease of testing and to minimise expense whilst maintaining the highest degree of control, the following equipment specification was used. Other specifications could produce a similar test environment, but the level of control would be limited as available UNIX based systems were shared resources; that is, other users and processes ran concurrently and may have affected results. Franklin
(1998) later suggested that response time in a UNIX environment to be even faster, so testing the slower DOS environment proved conservative.

<table>
<thead>
<tr>
<th>Client System</th>
<th>Server System</th>
</tr>
</thead>
<tbody>
<tr>
<td>60 MHz Pentium Processor PC with 4 MB RAM, a hard drive, 10 Base T network card, monitor, mouse and keyboard Microsoft Windows 3.11a (32 bit) SocketWrench © Software Visual Basic, version 3.0</td>
<td>60 MHz Pentium Processor PC with 4 MB RAM, a hard drive, 10 Base T network card, monitor, mouse and keyboard Microsoft Windows 3.11a (32 bit) SocketWrench © Software Visual Basic, version 3.0</td>
</tr>
</tbody>
</table>

Table 14: Test Apparatus for an Internet Echo client and server

Initially, both the client and server Echo shared the same computer. Procurement of a second computer enabled a 'conversation' between two machines using Microsoft Visual Basic. SocketWrench software can be obtained at this Internet address (http://www.catalyst.com). A description of the software source follows and sample software code for the server is included in the Appendix.

**TCPECHO:** This project is a more general implementation of the GENERIC sample program. It allows you to specify the remote host to connect to and the port number to use. A menu option can also be set which allows the program to act as an echo server as well as a client.

http://www.catalyst.com/

For comparison, an apparatus to examine the WWW protocol methodology in detail was set up, as the second of the two options mentioned previously. Discussion of the results in the Echo test with a supplier eventually led to a proposal for equipment from Buckinghamshire County Council, whom then provided the expense of time and equipment.

<table>
<thead>
<tr>
<th>Client System</th>
<th>Server System</th>
</tr>
</thead>
<tbody>
<tr>
<td>75 MHz Pentium Processor PC (Fujitsu) with a 8 MB RAM, hard drive, 10 Base T network card, monitor, mouse and keyboard Microsoft NT Workstation, version 3.51 Internet Browser (netscape, mosaic, and explorer)</td>
<td>133 MHz Pentium Processor PC (Vale) with 16 MB RAM, a hard drive, 10 Base T network card, monitor, mouse and keyboard Microsoft NT Server, version 3.51 WebSite © Internet Web Server</td>
</tr>
</tbody>
</table>

Table 15: Test Apparatus for an Internet WWW client and server

To set up this test, it was necessary to purchase and install an Internet web server software package on top of the operating system. The software program WebSite 1.0f was chosen, as it was the only known software available at the time (1995). Other products by Microsoft and Netscape used in later tests did not yet exist. This test also could have been done all on one machine; however, it was more clear what was happening if it was set up on two. Thus the Internet browser software was setup on the 'Client' computer and the server on the 'server' computer. Internet browsers installed on the university network also enabled tests with a different operating system (UNIX and Solaris), and on different types of computers (RISC based systems). It was also necessary to create a series of text files, using a text editor (such as Microsoft Notepad) and containing only data strings, which could be retrieved by any of the...
internet browser software programs. Finally, it was necessary to execute both the client and server software programs, so that the test could take place. Sample code exists at http://website.oreilly.com/.

The last test in this section of the thesis tested the ability for an apparatus to accept a command, and given this command, act accordingly. It was hypothesised that both the above apparatuses could be made to do this, but the web server apparatus was the more advanced of the two. Thus, of these two apparatus, only the second (WWW) was set up to see if this feature could be accomplished "usefully".

To do this, the same set up was used as in the previous test. This time however, Visual Basic was installed on the server system also. Using a Visual Basic software module provided freely by Denny (96) called CGI_Main, included in the purchase of WebSite, a test was set up and it compiled the same apparatus as used by Denny (96), duplicating his experiment.

6.1.2 Single and Multiple Database Journey Planners

Two suppliers of software and data for county councils provided examples single database journey planner software packages running under Windows 3.11. Both also provided databases as a separate file compatible with Microsoft Access 2.0. Both software packages were installed and executed. No tests or apparatuses were set up using these except that necessary to install and demonstrate their systems.

In the summer of 1996, a representative from one database supplier prototyped a multiple database journey planner, used by one of the county council's for their telephone enquiry service. Examination of these databases showed that a journey from the somewhere in the county to any point outside it but on the rail or coach network involved, quite naturally, an exchange point. This exchange point linked two common names for the same point, but listed differently in the two databases. Thus it could be said that the databases could now have a common exchange point between database tables. For instance, a bus trip might start the journey, end at a rail station, and then the journey would continue on the train. The alternative method as demonstrated by another supplier used the same concept, but only had one, much larger, database. Thus whilst no tests and apparatuses were necessary, this exercise demonstrated two methods of assembling a journey: either from combining three databases into one common format and thus one database, or having three databases with a means of getting between them.

6.1.3 Web Enabled Single Database Journey Planner

Having set up apparatus for the exchange of data using an Internet protocol to overcome hardware compatibility, and set up an apparatus to plan a journey, it is now interesting to combine the concepts in a further apparatus to see what the combined reaction might be.

In 1996 at two conferences, a system combining the concepts discussed so far was demonstrated. These tests used the same equipment as in the Test Apparatus for an Internet WWW client and server, but
included Microsoft Access 2.0 installed on the server machine also. Using the methodology provided by WebSite, the same apparatus as above was set up again. Instead of accessing the sample database provided however, a number of queries to test how might a journey be accessed from the database via the browser and web server and a script were set up.

To do this, a number of queries within the database containing the journeys were set up. These queries included the date and time as parameters, as well as the origin and destination. The apparatus was set up with a fixed list of origin and destination codes for this experiment, as well as a fixed time and date, known to produce a result under the scenario above which did not include any WWW software.

Using the same Visual Basic software, and modifications to the CGI_Main software tested earlier, a test of this concept was prepared. A script was created, which when retrieved, provided an opening HTML page as before. This page included a SUBMIT command button. To run the test, the user needed to simply click on the "SUBMIT" button. A software program, the script, then included instructions to send the assigned origin, destination, date and time parameters to the database query, execute the query, and then send the results back through the script, to the WWW server, which then sent the text string back to the user, to be displayed on the browser software.

The test data was an origin of Hanslope, and a destination of Hammersham. The date and time were that of the system clock. The results of this test were demonstrated successfully at a conference in the summer of 1996, before industrial suppliers then incorporated these concepts into their own demonstrations. One of the database suppliers then set up a commercial service on the web for one county, and this was followed by other counties, both by that supplier and by other suppliers.

6.2 Homogeneous System

In 1997, an apparatus was set up that combined all concepts so far, except to have only one supplier. Effectively, it would be a homogeneous, multiple journey planner, public transport web navigator. Any supplier would do so long as they had more than one database. From existing contacts, one supplier volunteered to trial the concept. However, being completely internal, the supplier needed to be convinced and supervised in the creation of an out-of-house apparatus, whilst ensuring that the apparatus would indeed do as required.

Specifications were drawn up for the prototype described here. Several meetings with suppliers and users as represented by county councils suggested and established a collection of keywords and an initial pool of definitions, drawn from work by EUROBUS, DATEX, and later, TCIP, (see Chapter Two).

As stated in Chapter Four, the idea of passing longer data strings in a method analogous to what is now known as eXtended Markup Language (XML) was conceived in the autumn of 1996. However, the first drafts of XML were independently published on the WWW in November of 1996.
<table>
<thead>
<tr>
<th>Client System</th>
<th>Server Systems (3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>75 MHz Pentium Processor PC (Fujitsu) with a 8 MB RAM, hard drive, 10 Base T network card, monitor, mouse and keyboard</td>
<td>133 MHz Pentium Processor PC (Vale) with 16 MB RAM, a hard drive, 10 Base T network card, monitor, mouse and keyboard</td>
</tr>
<tr>
<td>Microsoft NT Workstation, version 3.51</td>
<td>Microsoft NT Server, version 3.51</td>
</tr>
<tr>
<td>Internet Browser (netscape, mosaic, and explorer)</td>
<td>Internet Information Web Server</td>
</tr>
</tbody>
</table>

Table 16: Test Apparatus for an Internet WWW client and server

This apparatus required special software to be written to combine various technologies, allowing a data string from one application to pass through a second application and into a third application. This forwarding of data strings created a transparent method of exchanging data between different operating systems and often proprietary search engines of local databases containing public transport journey information.

In the latter part of September 1998, a system for an internet based, multiple database journey planner was set up at one of the suppliers to demonstrate a cross boundary journey using the methods described in this thesis. For the demonstration, the particular systems involved included the following:

- a local bus journey from a bus stop origin in Amersham or a selection of nearby areas to a national rail exchange point in Buckinghamshire on the national rail network,
- from this national rail exchange point to a pair of national rail exchange points in Guildford or a single national rail exchange point in Brighton, and
- from either this pair of national rail exchange points in Guildford or the single national rail exchange point in Brighton to a final destination in Brighton or Guildford on the bus network.

This demonstration involved a system of three public transport journey planners from one company. The demonstration set out to show the ability to handle more than one local exchange point, a critical feature in realistic journey planning. It sought to demonstrate that the journey could be requested in reverse.

6.3 Design Analysis for Heterogeneous System

To demonstrate the interconnection or interoperability feature of the extensible model, it was necessary to have both an "active" and a "passive" journey planner from more than one industrial partner. By definition, a "passive" journey planner is one that simply listens for requests and responds with the requested information. The "active" journey planner is the one that interacts with the user, and may make requests of one or more "passive" systems to formulate an integrated selection of itineraries for the traveller. A second company agreed to provide a journey planner for testing a heterogeneous system with multiple journey planners and multiple occurrences of at least one public transport web navigator. Each of these journey planners needed to allow journeys across database boundaries from one entry point.
Several design decisions to set up a test apparatus meeting all criteria had to be made. Due to the comprehensive nature of this company's national rail journey planner, it was decided to use their journey planner for the national (rail) mode carrier in place of the more limited national rail journey planner provided by first supplier. Thus a heterogeneous system, with respect to mode and supplier, could be demonstrated effectively. Further to this, the data supplier for the national coach company provided and set up a working journey planner to serve as the second national mode. Two county "local transport network databases" (mainly bus), neither adjacent nor adjoining, existed for the respective local connections at the poles of any journeys. These journey planners effectively made up the passive routeing engine components.

**Figure 21 : The Heterogeneous Test System**

![Diagram of the test system](image)

A common component was necessary too. For instance, as discussed in the Third and Fifth Chapter, a common index was necessary to link stop points, communities, journeys, and database sources. Given the national, UK nature this research had now assumed, this component was named the "national gazetteer". For this component, this study obtained a copy and research licence of the database from the Office of National Statistics.

It can be seen from the diagram of the test system (Figure 21) that only parts of the system were available for testing. These parts are greyed slightly to provide emphasis. It is further emphasised that the data within the system is sparse, so much so that comprehensive testing at a system level may not be possible...
unless a concerted effort is made nationally by a significant proportion of the county councils to improve the data.

For instance, even “timing point” level data is not available in some areas, and its descriptors are often vague. More precise data is necessary for a comprehensive evaluation of the system. However, setting these limitations aside, it is still possible to determine some information and analyse it at lower levels in the system, even if it is not as rigorous at the top level as may be available in the future (presuming national adoption of the principles involved).

6.3.1 Hardware and Software

To test the concepts, several software applications were necessary. Three personal computers with 128 MB of RAM and at least a Pentium 300 MHz processor with IDE hard drives, network card, and standard keyboard, mouse, monitor et cetera were constructed as servers from bits used in previous experiments. The national coach supplier provided a fourth server. A Microsoft NT server operating system, Version 4.0, was installed on each server. It was necessary to use Service Pack 4, as this service pack resolved various issues with interactions between the WinSock (i.e. the TCP/IP stack application programming interface for Microsoft operating system) and various dependent software programs. Next, Microsoft Internet Information Server, Version 3.0 or higher was also installed on each of these stand-alone servers. All servers had a network connection to a local hub at 10 Megabits/sec using a “ten-base-T” twisted pair Ethernet. Each of the servers was then registered with the DNS. It is salient that these options were chosen out of practicality, and not necessarily based on any specific criteria. The test system could have been Unix based, and any brand of web server.

To facilitate the exchange of data between systems, a selection of off-the-shelf software “modules”, where a module is a collection of subroutines, were combined to write software programmes called “scripts”. Unlike many existing software programmes which are activated or “executed” by clicking on an “icon” or by typing their name on a “command line” and pressing “ENTER”, scripts can be executed using the GET or POST method in a Hyper Text Transfer Protocol (WWW) environment. This execution is accomplished by pointing a Uniform Resource Locator (URL) link to the address of the script and attaching the necessary data string (in the correct format) within an Internet browser such as Mosaic, Opera, Netscape, Explorer, or Lynx. As the data strings were quite long (often above 255 characters), a decision was taken to use the POST command exclusively.

Several combinations of software modules by several suppliers could provide the necessary effect receiving a data string and returning a data string that had been processed. Several choices existed in practice: Common Gateway Interface (CGI), WSAPI or ISAPI (Website or Information Sever Application Program Interface), and Active Server Pages (ASP). Of the four, the latter three are compatible with the first but not vice versa. ISAPI, WSAPI, and ASP use what are called Dynamic Linked Library (DLL) or ActiveX scripting to process and execute a series of commands that receive, parse, translate, manipulate, calculate, and return a data string. To use CGI with Microsoft Internet
Information Server, only one alternative was found; it is a software module called CGI4VB, a freeware module by Kevin O'Brien, in conjunction with another freeware module by Bob Denny called CGI_Main. It was also necessary to use Winlnet.dll or Winlnet.bas, both used by Microsoft's Internet Transfer Control Protocol, in conjunction with these modules, to provide the required functionality.

Other applications by other software houses are available, such as the SocketWrench by Catalyst Software and dsSocket by Dolphin Systems, or a combination of applications by Borland using Delphi or Delphi C. Any of these combinations of software modules could work suitably, but certain of these have more features than others. For instance, SocketWrench and dsSocket both allow asynchronous methods of internet communications over the seven layered sockets protocol. The significance of this is discussed in Chapter Seven and further information and sample code is available.

6.3.2 Relationships and Passive Commands

Several relationships between the routing engine databases and the national gazetteer were necessary also. An "active journey planner software" that functions like a broker has been suggested. In example, it talks with the passive journey planners and databases, and it talks with the traveller. To achieve this brokering, two sub-processes were set up: one to define the journey, and one to find a solution. However, it is absolutely crucial to remember that in the research described here, whilst only one active system was demonstrated, a number of instances of one or more active systems could be executed at the same time, and these need not be of the same brand or software supplier. Thus a number of companies could supply these systems. To encourage this, examples of the messages and schemas necessary are included in Appendix E.

6.3.2.1 Passive System Messaging

A passive system is also necessary. Following the review of messages in Chapter Four, a modification of HTML as a messaging structure was decided on. Besides the HEAD command, two methods exist to send and obtain data within HTTP requests: GET and POST. Either method is equally acceptable according to HTTP standards. These same standards state as a convention however that the GET method is used primarily to retrieve data, whereas the POST method is used to update, delete, or create new information. Requests for places do not update or change date, but simply make copies of the data. Secondly, as the quantity of physical data being exchanged to pose the travel request is relatively small, i.e. about 50 characters, (i.e. under 255 characters) it is prudent to use the GET method in sending gazetteer requests. However, the POST request method was adopted in order to have consistency. Within these standard HTML tags, what are now known as XML tags and data were added.

Given this messaging structure, three message "dialogues" to pass data strings were tested initially, according to function. These message types were translated into "commands" and set out in a common

---

12 Dr. Anthony Lock helped write an in-house collection of modules also using WININET.
“protocol”. The first command was “PLACE”, which, given a community, returned a list of names representing bus stop points and their unique identifiers. The second command was “EXCHANGE”. The exchange command took as its argument a PLACE, and thus returned a list of exchange points for that particular place. Exchange points are in fact places themselves, but have a dual classification, as they can either be an exchange point or an origin/destination. Finally, a command was set up using “SEARCH”. The purpose of the search command was to enable a remote journey planner to return a list of journeys given a particular journey definition. Data strings were then attached to the commands embedding data using XML (see Appendix E).

6.3.3 Design Analysis of Request for Places

6.3.3.1 Purpose and Method of Sending a Request

Previous sections defined a hierarchy of country, county, village, and place. Having decided to break the hierarchy at the village level, the purpose of the request then becomes one of retrieving a list of specific places (or some equivalent form of data) within the chosen village. This retrieval of data allows the traveller to further refine the selection of an origin or destination.

6.3.3.2 Formats of Request Types

Journey planner systems might request data in several different ways. For instance, the requesting system may have a map with co-ordinates and wish simply to have the remote system return all places within a rectangular or circular shaped area bounded by a set of co-ordinate pairs. Alternatively, the requesting system may have a postal or zip code, and request all places within that postal zone. Depending on the accuracy and length of the code, this could be a large area of several kilometres or just a few houses on a street. Further, the requesting system might like to request places within a particular area, all places in an area matching a string of letters, or some combination of both. Ideally, the receiving system should be able to handle any requests such as these. However, because of practical limitations with possibilities already discussed, it was best to implement options, as data became more readily available.

The lack of comprehensive map data for the UK with rights to be used freely on the internet, and the lack of geo-coded location data, such as latitude and longitude, in existing transport database systems, led to an initial focus on the textual matching of village names in county databases. Therefore the first implemented request was a POST statement which requested all places within a particular village.
6.3.4 Design Analysis of Requests for Exchange Points

6.3.4.1 Types of Requests and the Expected Response

Three basic scenarios result in three types of exchange point requests. Using the example above, one case is where a Hampshire journey planner makes a remote request to the Buckinghamshire journey planner for National Exchange points for the place in Buckinghamshire. Another case would be if the destination were in an adjacent county, such as Surrey, then a Hampshire journey planner would request from Surrey adjacent exchange points, and perhaps the local exchange points as well.

Thus, three types of exchange points are necessary: local, adjacent, and national/international. Local exchange points are typically not exchanged with remote systems except in some situations involving adjacent counties. The request has the same structure generally as the other two. It follows the structure of “given this point or place, please return all local exchange points”. Since transport services for local trips are typically not held in a separate database, it is not usually necessary to return the location of the journey planner service. However, in the case of requests for adjacent county and national exchange points, it may be necessary to return the location of the journey planner covering these transport services.

6.3.4.2 Formats for Requests

Even so, several design decisions are necessary as the data may be requested in several different ways. In all cases, it is necessary to determine the type of exchange points that will be requested. A standard definition is then proposed simply as LOCAL, ADJACENT or NATIONAL. A second consideration is then how to define the point requiring exchange points.

One method is to define the village and then its place as it was sent via the gazetteer from the remote system. It would normally be necessary to include both the village and the place in this case. For example, the request statement might include “village=value&place=value”. This methodology might still be ambiguous if two places had the same name within a village. The decision has already been taken to define all villages uniquely within an administrative area.

However, this might present too many ambiguities and restrict later flexibility. Thus it may be better to define the point in the following manner:

“place=<county>bucks</county><village>Amersham</village><place>Rail+Station</place>”

This latter method may seem more cumbersome, but it does allow the flexibility to define a place as:

“place=<latitude>value</latitude><longitude>value</longitude>"
This could allow for various alternatives to the method of defining the structure of the statement, without needing to redefine the format.

A final issue is whether or not the requester wishes to limit the number of exchange points returned. Normally this would not be the case, but some ordering or ranking could be accommodated and thus the supplier of the exchange points may wish to provide many more exchange points that the requester may have use for. Thus, the possibility to restrict the number of exchange points has been introduced.

6.3.5 Design Analysis of Requests for Journeys

Chapter Five proposed and developed three possible methods of implementing the request for journeys or "connections" as EU-Spirit seems to call these elements. This research chose the verbose method.

However, for reasons of transmission, only one journey solution was allowed to be retrieved at once. It was preferred that if there were to be multiple requests, that these are queued! The reason for this was largely due to preference of one of the suppliers, and their ability to then marshal the requests between separate processes and queue them more efficiently within their internal software subsystem. It also allows the HTTP Server to handle the queuing and throttling of the requests, making the scripting software more simple in function and easier to program, and thus faster and more efficient.

For instance, instead of taking five requests and bundling them together and sending one message which is then received, processed, answered and returned, the five requests were sent sequentially (either synchronously or asynchronously, as explained in Chapter 7). In this manner, a router could intercept the messages and send every other one to another server or process. Where the number of requests exceeded the processing time, these subsequent messages were queued. This had the benefit of allowing the first lot to return whilst processing the second lot and so forth, so that a long time is not spent waiting for a response which might be damaged in transmission.

6.4 Interactions between Active and Passive Software Modules

The purpose of the enquiry engine is to help the traveller define an accurate journey. This definition is passed to Routeing Engine, which, without further assistance from the traveller, determines a range of alternative solutions for the traveller, and displays them.

6.4.1 Enquiry Engine

Initially, a traveller needs to enter the system. Thus an "Initial Point of Entry" or portal into the system is necessary. This portal could be anywhere, and take in any number of forms or brands. However, a bland
and generic system for testing was sought, leaving the more sophisticated portals for the industrial partners that joined the research in 1997.

Based on the solid precedent from the survey of existing systems, it was assumed that the first action by the traveller would be to define the parameters for the journey. Thus, the processes described here send the traveller straight to the Enquiry Engine with the initial execution of the program, be it by a web browser or a command-line executable programme. The initial code then simply displays a series of text input fields, and a drop down list on an internet "form". The origin and destination are normal text boxes.

The Date is a forward-looking drop down list of the next 14 days, and the time is either a drop-down list of times or a free entry text box. The final parameter in this system allows the traveller to choose whether this time was the departure time from the origin for arrival time at the destination. The figure below shows a sample of the executed code in a browser. The reason for using the conventions chosen was simply to follow existing conventions as used by the sponsoring authorities, or to follow broad trends. Other methods are of course possible. It is important at this point to state that the preference to use a browser was just that, a preference. Any Windows, DOS, Apple, OS/2 or UNIX program that executed later scripts as a collection of sub-processes could have been chosen.

Figure 22: Sample image showing Initial Point of Entry screen
Having this screen in front of them, the traveller is obliged to enter some text characters for the origin and destination. The traveller also needs to select a date and time, and choose whether the journey time will be the departure time from the origin or the arrival time at the destination. These entries are then validated against ready-made criteria.

It is evident that an incremental approach to these criteria lists is necessary, and for practical reasons, the research described in this thesis concentrated on the villages in parts of Hampshire, Buckinghamshire and Surrey to start. As a future measure of the extensibility of this research, other areas could be included.

It is relevant to note that having compiled a list of villages, an obstacle exists which may be unique to the United Kingdom. No freely available list of village names or any sub-set, is available for “publishing” on the World Wide Web. This restriction results from licensing agreements and the need for government agencies to recover their costs. In the United States, the government publishes a gazetteer of American villages on the World Wide Web.

6.4.1.1 Getting the Internet Address

It proved practical to prototype a gazetteer for the United Kingdom with a political hierarchy down to a village or community level before splitting it. This was because all suppliers have their own names for a place, and ownership of the name often rested with the county or transport service operator involved. As this name is changed, this updating can then be done by the service or county involved.

This limits the selection of “places” to about 50,000 villages. It was then more practical to place individual bus stops, stations and so forth within a particular village. **However, each village name within a county then had to be absolutely unique.** This raises several issues relating to the creation of the original list of these village names and their storage and updating (addressed in later research).

After the entries have been validated, the parameters in the origin and destination are compared with a selection of locations in the “National Gazetteer”. For example, if the traveller enters “Sou” in the Origin text box, this system chose to display a list of all communities from within the national gazetteer which began with the letters “Sou”, either at the start of the place name (“Southampton”) or at the start of the second word (“Old Southsea”). It could have chosen to show the communities starting with “Sou” in the first word however. The same comparison exists for the destination, also returning a list.

6.4.1.2 Getting the Stop Points

Once the traveller has selected an origin and a destination community successfully, this origin and destination community is fed back to the enquiry engine with the date, time and depart or arrive by parameter. It was then proposed to query the national gazetteer for the address of the data resource which contained further information on that community. For instance, this particular test methodology suggests
that it is necessary to pick a precise start and stop point so as to have as precise a journey definition as possible. Precedent from previous journey planners did not necessarily agree with this philosophy.

In the research described in this thesis, it was decided to associate the addresses for the data sources to the administrative area responsible for providing journey-planning services for that area. As each community has an administrative area defined to it, it is possible from a relational query to then determine the address of the relevant data source. This address takes the form of a URL containing the address of a web site script and the necessary parameters. An example might be “http://www.journeyweb.org.uk/scripts/pjp.exe/su”.

Using the first of the commands defined earlier “PLACE”, it was chosen to programme the enquiry engine to feed the community name identifier (i.e. “E0000372”) to the data source responsible for that community’s administrative area. The passive journey engine, described in a previous section, checks to see that it recognises the community. Should it do so, it then checks for a list of possible stops within that community or associated with that community. Should the list contain at least one record, this list is then returned via the standard messaging shown in the Appendix, with respect to the passive journey planner. Should the list be empty, a single message set is returned without any records.

This process is repeated twice. Once for the origin and it is repeated again for the destination. The enquiry engine then has a list of stops for the origin and destination. It returns these lists to the traveller whom must now choose a particular start and stop point. At this point, the software used in this research then receives the parameters and communication with the traveller ceases until the final response is received.

6.4.2 Routing Engine

All the data for a particular administrative area is normally held locally. Further, the case may arise whereby the connections with regionally adjacent administrative areas are of sufficient quality to allow reasonable responses for regional information. However, once one exceeds a particular distance, the response time for a particular request would suggest that the enquiry be dealt with a journey planner with at least one local point. This is an issue requiring greater definition in the protocol, discussed in later chapters.

The first task for the routing engine is to extract the journey definition from the traveller request. It is then necessary to extract the respective villages from the journey definition. Using the village as the keyword, it is possible to determine the relevant local governmental administrative area for both the origin and the destination. It is at this point that several possibilities arise.

• the origin and destination are both in the same database area
• the origin is in one database area and the destination is in an adjoining database area
• both the origin and the destination are in non-adjointing database areas
Within the second possibility lay the more complex cases of what happens if the remote county is in an adjacent county. These sub-cases can be dealt with separately. This is due to the nature of the databases containing trips originating in the local county but which terminate outside the local county.

6.4.2.1 Local journeys

The 'Local Journey' is the simplest case to deal with. As the origin and destination are both in the same database area, the value for names are simply passed in the form of the agreed protocol, regardless if the particular database is "local" or not. The journey planner then makes a standard request between the origin and destination of the relevant database software supplier at the appropriate internet address without the need for exchange points.

6.4.2.2 Journeys between two National stations

This case of a journey between two rail stations, coach stations or airports is also straightforward. In this case also, values are simply passed in the form of the agreed protocol, straight to the supplier of national data between those two points. The journey planner then makes a standard request between the origin and destination without the need for exchange points. However, should the case arise that no journeys are available between two rail stations for instance, the system may be obliged to provide alternative journey solutions via another mode (perhaps coach or air), if available, as a courtesy or should regulations or competition require it. Whilst these cases are more complicated programmatically, research already underway is investigating the feasibility of such requirements.

6.4.2.3 Local Journeys to a National station

This case allows the journey planner to present solutions to journeys between a local point and a national mode point such as a rail station, coach station, airport, or ferry terminal. In the instance of a local origin such as pub or street intersection, the journey planner needs to request the appropriate exchange points relative to the mode of the destination, or comparable mode if it is to provide an alternative as in the previous paragraph. The journey planner then marries up the journeys between the origin and the exchange points, and journeys between exchange points and the destination.

6.4.2.4 Mid Distance Journeys

This class of journeys are the so-called 'adjacent county problems' and are the most difficult to establish and define, and as such are not fully developed in the research described in this thesis. However, some simple descriptions set the foundations for further research already underway.

6.4.2.4.1 Local to Elsewhere – Adjacent and Adjoining

This journey is where the origin and the destination lay in the same database area, but different political boundaries as defined by the gazetteer. In this case, the origin may be said to be in the 'primary' database and the destination in the 'secondary' database of the adjacent or adjoining (these having been defined
elsewhere in the thesis) county. As the address for the secondary database is the same as the primary database, the journey planner can thus determine that a journey may reside wholly within the primary database. Thus, the journey planner makes a standard request between the origin and destination without the need for exchange points. The inverse of this follows a similar methodology.

6.4.2.4.2 Database areas with different levels of detail beyond the area boundary

Of concern also is the case where data in the adjacent or adjoining county is of a different level of detail than that in the primary database area. For instance, it may be that one database area, say London, holds information that is precise and accurate down to the nearest square metre. In speaking with a journey planner in an area where such resolution may not be deemed necessary (such as low-density areas), a conflict in naming or implied name and corresponding resolution could evolve. In this case the journey planner is proposed to examine the properties, if they are available, of the opposing end of the journey to determine if the remote journey planner can determine a solution to the journey enquiry wholly from data within its domain. However, as these processes require further refinement, solutions are not presented here.

6.4.2.5 National Journeys

A long distance journey is defined as a local journey at each end of a 'long distance' journey by one of the national modes (air, rail, coach, or ferry). In this case the journey planner has to do the following:
1. determine the origin and destination
2. decide on the appropriate local exchange points
3. contact the local journey planner for the origin and destination and get relevant exchange points for both
4. contact the coach/rail/air operators and obtain information about trips between these exchange points
5. contact the local journey planner for the origin and destination and get journeys between the relevant exchange points for the destination and the relevant exchange points and the origin
6. sort out the 'best' route or routes

The journey planning system is symmetrical and, as such, either the origin, the destination, or both can be 'remote'.

6.5 Summary

Looking back to the objectives, it is possible to highlight what has been achieved thus far, given the very limited quantity of data and journey planners available. In the objectives, a number of functions were listed. These functions are repeated in the following table. Subsequent columns identify at what stage these functions were achieved.
To reiterate, the system described in section 6.1.2 pasted together a rail, a coach and a county database from one supplier. Thus it allowed journeys with multiple modes and it crossed political boundaries. It did not however provide detail information regarding the end of the journey that resided outside the county boundary. Whilst it did function with areas missing, it was necessary for it to be aware that there were other areas to meet this criteria. The system described in section 6.1.3 enabled access via the Internet enabling the system independence of computer platform at the client end. In section 6.2, the system separated the functions of journey enquiry and journey response more clearly by introducing an independent gazetteer. It also enabled journey planners of the same supplier to interrogate each other.

<table>
<thead>
<tr>
<th>Test Function</th>
<th>6.1.2</th>
<th>6.1.3</th>
<th>6.2</th>
<th>6.3 &amp; 6.4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expand journey planning to include other modes</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Across political boundaries</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Independent of hardware and software platforms</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Allow journey planning to function with areas missing</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Allow journey planners to interrogate each other</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Independent of private database protocols</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 17: Functions of tested systems

As can be seen from the table, only the journey planning system described in sections 6.3 and 6.4 can be said to have all the characteristics set out in the objectives. As such, only that system can be said to be extensible as defined thus far.

In summary, this chapter discussed the various hardware and software of the initial systems, as well as the final system. The detail within it for the different processes, software, and hardware necessary to reproduce the experiments discussed in the following chapter was laid out. It also described in detail the different types of journeys. It is important to note that whilst the software, hardware, and algorithmic processes and protocols are well defined, a major gap exists in that the data is not at a sufficiently comprehensive level or coverage broad enough to permit rigorous testing of all potential aspects of the developed system. Nevertheless, it is possible to examine fundamental characteristics of the system affecting the extensibility of the system; these findings might benefit the design of navigators and database system implementations should the research be utilised in a national system.
Chapter 7 - Results and Evaluation

7.1 Evaluation of Extensible Model

This chapter analyses and evaluates the apparatus described in the previous chapter. The first section of this chapter briefly reviews the decisions made with respect to the architecture, messaging, the gazetteer and journey planning or navigation, and discusses some of the reasoning behind them. The second section discusses some of the possible tests that could be done in an ideal situation, and their context relative to this research. Where possible, experiments were run and results obtained, but some tests will, by nature of the available data and resources, be more rigorous than other tests. In conjunction with presenting the test results, this section analyses the impacts and some of the consequences, providing an indication of future behaviour. It may also show where future research might fill remaining gaps in the knowledge in data dependent areas.

7.1.1 System Architecture

Of the system architectures reviewed, it is recommended that a distributed system architecture be used, so as to have a globally extensible system. Unlike the warehouse approach, it could encourage increased market competition in the supply of software between and within different layers of the system. In a different political climate, this level of complexity might not have been necessary, except for the global applicability of this research. Whilst it is not possible to determine a definitive consequence, results presented in later sections may be able to provide an indication of the relative merits of this decision.

7.1.2 Effects of Message Format Decisions

The field structure can be implemented in different combinations. Of the different field structures examined, a more "verbose" (i.e. the identifier tags contain more characters) tag naming method is recommended. However, it is obvious that if the structure is verbose, there is a measurable impact on the time necessary to transfer the request across the Internet. Later analyses may provide an indication of the relative importance of this issue with respect to the other aspects of the system performance.

With respect to packaging requests together to reduce response time, it may be shown in this chapter that the network time as a percentage of the total response time is small. If this is the case, the relative merits of doing so may not be worthwhile, as the additional overhead in sending multiple small packets verses one big packet is small. Further, any benefits might be counter-balanced by the time spent processing the multiple requests. The examination of the relative times to perform various activities may give further indication of the relative merits.
7.1.3 Implications of Design Decisions on Gazetteering

Splitting the gazetteer at the village level has several implications. First, the question arises as to who has responsibility for the creation and maintenance of the gazetteer and the relationships within it. Splitting the gazetteer at the village level also has profound impacts on the structure of the 'registry' (i.e. tables of exchange points) and on the search process for journey solutions. These impacts ultimately affect journey solution response time.

Imparting relationships within the gazetteer is an unresolved group of issues. Is the responsibility for data shared or tendered to one organisation? Who has rights to what data? Whose data is correct? In practice, county councils normally provide journey planner services to cover the whole of an area. Generally in the United Kingdom, the responsibility for data for a county is contracted in whole to one data-supplier company. Thus, most companies organise their data by county, with some notable exceptions, and it is possible as a first iteration to directly link all the villages in an administrative area to one data supplier.

This linkage is important as it forms the basis for finding places, exchange points, and journeys, and has several implications on searching for exchange points in the registry and searching for journey solutions. For instance, exchange points may no longer be simply defined, but must be classified into local, adjacent, and national/international exchange points. Also, no longer can searching be done by what is know as a "military method", i.e. all points are possible exchange points. Journeys now have a hierarchy resulting from the decision to make distinctions as to whether a journey is local to a county or not.

The following scenarios of the potential implications illustrate the importance of this issue of referencing. Presume for instance that the definition of the gazetteer contains only one reference for each county. If it arises that the server referenced does not exist, is not functioning, or is overloaded, all information for that county disappears from the list of available resources. So as a result, the higher in the hierarchy the relationship to the "more refined detail" is, the more data that will be potentially lost if anything happens.

Similarly, if the gazetteer contains only one reference for a village, and that reference disappears for some reason, all information referenced for that village disappears. However, it is likely that several counties hold data on any one village, even though only one would be the primary provider of data for that village. It is also necessary to allow multiple data service providers to be listed for the same village. As it is lower in the hierarchy, (i.e. at village level instead of county level, with a county being much larger than a village), and some overlap exists at village level, the potential impact is reduced.

However, it is not yet clear where to hold the relationship with respect to the exchange points. The primary reason for this is that if the boundary of village is split by a large obstacle such as a motorway, railway line, river or hill/crag, then it could be that different sides of the village or town will gravitate towards different exchange points logically. Similarly, railway lines were not always built in logical directions, and some lines compete for very similar areas, but may be separated by the type of clients.
For instance, on some days of the year in the era of building railways, horse racing or similar events drew sufficient crowds to warrant building competing railway lines. Whilst not universal across Britain, this exception must be catered for in methods of the allocating exchange points.

Thus, to limit complexity whilst maintaining adequate flexibility, the definitions of relationships and the types of relationships have been defined as follows:

<table>
<thead>
<tr>
<th>Relationship</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local Government Area to Locality</td>
<td>One to Many</td>
</tr>
<tr>
<td>Locality to Transport Network Stop</td>
<td>Many to Many</td>
</tr>
<tr>
<td>County (Village) to Data Supplier Address</td>
<td>Many to One (Primary)</td>
</tr>
<tr>
<td>Data Supplier Address to County (Village)</td>
<td>One to Many (Secondary)</td>
</tr>
</tbody>
</table>

Table 18: Data Relationships

These relationships would not necessarily be optimal for all countries because of different political boundaries or market practices (i.e. a state in America might wish to tender individual regions of a state to separate transportation information suppliers). Thus the decision to split the hierarchy at the village level may not have been optimal, and thus this issue is also unresolved. Therefore in making design decisions, it may be necessary to consider the local market mechanisms and related political boundaries before adopting what might be a country specific solution. The structure of the definitions should still hold however, even if more definitions might be necessary for a given political area.

7.1.4 Effects of Design Decisions on Navigation

Several implications affect navigation also. As the navigation depends on the pasting together of journeys using exchange points, it is self evident that the accuracy of the place name and registry entries is crucial; as accuracy increases, so will the usefulness of the itinerary. An error in an exchange point reference entry can have serious consequences on journeys involving related points. From a practical perspective, this accuracy can only be achieved with local knowledge, and therefore this is one of the primary reasons for insisting that the gazetteers and exchange point tables be maintained by the local government authority geographically closest to that point. In England, this is usually the county council but elsewhere may be another governmental or religious body.

Should a reference relating the supplier’s name to the point be included with the response for exchange points? For instance, does one expect that in returning an exchange point, that one should expect to see the name of the supplying database responsible for that particular point? Or does one presume that if the point is a rail station, for example, that one company is responsible for all rail stations and provides schedule information for all services through that station? In this research, it proved advantageous to provide the name of the supplier with the exchange point, as these presumptions were not always true. In exchanging the name of an exchange point across between journey planner systems, it may be that a fuller definition would be required, at least to county level.
7.2 Context of the Analyses

This section explains what is tested and why. In the introductory chapter, this research states that the necessary functions of an extensible journey planner model would be to function across political boundaries, include multiple modes, allow interoperability between different operating systems, allow mutual interrogation between journey planners of different suppliers, function with areas missing, and be independent of the database systems which underlie the proprietary journey planner software. It also sought to be a mass enquiry system (i.e. thousands of individualised enquires), and a mass access system (i.e. price per unit enquiry very cheap/free to user).

It is recognised that it is not possible to prove or rigorously defend the supposition that the proposed model meets each of these with the available resources. Even if the resources were available, many of the decisions made here would only be justified as a function of cost from a particular perspective.

For instance, the decision to make it a distributed system lay in the accuracy of the data, not in any particular technological advantage gained. The decision for using the messaging structures suggested would perhaps only be justifiable from the perspective of lowering the cost of implementation, as other technologies may be better in some aspects, but cost significantly more money to train the staff. Whether a particular methodology suggested is “best” is perhaps more qualitative than quantitative and as such does not lend itself to the rigors suggested for this work. These issues are more important from the perspective of costing the technical implementation of the overall system. However, some of the tests can provide an indication of the relative consequences of these decisions.

Thus, the data created and analysed in this pioneering research is aimed at benchmarking a set of references from which future (applied) research results can be compared. It is salient to state that at the time of testing, very little schedule data existed, and even less existed in high enough quality to be useful, so it is necessary to be aware that the interpretation of results is limited by certain caveats.

In benchmarking a set of references, and given that very little schedule data exists, perhaps the next task is to clarify what these references are, and how they might be used. If one returns to the traveller requirements, several underlying issues emerge. At the risk of confusing the reader, these issues are grouped under the headings of performance, logistic and accuracy.

7.2.1 Description of the Layers in Performance Analyses

Measurement of the technical performance can be grouped by introducing several “layers”. Within each of the layers are the underlying sockets, containing seven internet “sub-layers”. For this research, these sub-layers are normally treated as one unit.
On the top layer is any number of hardware devices with built-in software clients. These devices might be a phone, a watch, or desktop software such as a browser or an internet enabled application. These devices communicate with a choice of navigators or journey planners, which all interact with a "National Gazetteer". These navigators contain the intelligent navigation software and either have a copy, or are directly connected to, the associated “collective” geo-spatial databases. Thus this top layer is between the traveller's device and the connected navigator.

<table>
<thead>
<tr>
<th>Layer</th>
<th>Layer</th>
<th>Expected Response Time</th>
<th>What to Vary</th>
<th>Why Measure this</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internet Enabled Client Device</td>
<td>Active Navigation Internet Server</td>
<td>up to 5 sec</td>
<td>Vary the type of device (watch, phone, TV, PC, etc.)</td>
<td>Measure the difference between this response time and that of the cumulative subsystem to determine the relative affect of transmission to different devices has on response time</td>
</tr>
<tr>
<td>Active Navigation Internet Server</td>
<td>Passive Routing Internet Server</td>
<td>about 0.5 sec</td>
<td>Measure the difference between this response time and that of the cumulative subsystem.</td>
<td>Measure Affect of Internet Overhead</td>
</tr>
<tr>
<td>Passive Routing Internet Server</td>
<td>Proprietary Engine</td>
<td>about 0.5 sec</td>
<td>Measure the difference between this response time and that of the cumulative subsystem.</td>
<td>Measure Affect of Additional Software Layer</td>
</tr>
<tr>
<td>Proprietary Engine</td>
<td>Schedule Data for Individual Area</td>
<td>up to 30 sec</td>
<td>Vary the operating system and Programming language used</td>
<td>Measure influence of different operating systems (mac, IBM, Unix, NT), and different Internet server software packages (Apache, Netscape, Microsoft, &amp; Website)</td>
</tr>
</tbody>
</table>

Table 19: Tests showing the Relative Impact on Performance

The layer below that is what is called the passive layer in this thesis and is sometimes referred to as a servant. It contains software applications called “servers”, which sit and listen for requests. Thus this layer is between the active navigators and the passive servers.

When a passive server receives a request, it dissects it and passes the relevant information to the proprietary engine. Depending on loading however, it may first act as a router to send the request to one or more proprietary engines. Implicit in the need for this is the suggestion that the proprietary engine is indeed the slowest component in the system for some types of enquiries.

The “bottom” layer contains the data reservoir (i.e. the timetables) and the engine that searches through it. Within this layer is a small matrix of possibilities combining hardware, operating systems, and proprietary software journey scheduling engines whose combined behaviour may vary measurably from other configurations and groupings.

7.2.2 Merits of Performance Measurement

Depending on the final application, particular behaviour by any one component of any subsystem could have a measurable impact on the design and response of a dependent software sub system at a higher
level. Considerable merits could exist in using one configuration and grouping over another similar grouping. In a laboratory environment, it should be possible to provide an indication of the relative effect of the internet, the implications of a distributed design, and hypothesise potential response times at the navigator level for certain configurations and groupings described later by measuring the absolute time at any particular point.

An example at between the bottom layer and the passive layer illustrates this point first. It would be interesting to see the difference if any between the response times of particular dependent components given a variance in the operating system and the internet service software which resides on the desktop. For instance, is there a particular combination of operating systems and Internet Server software which performs the given tasks necessary in this research more quickly or reliably? In adding the proprietary engine to the matrix, is some combination of engine, internet server, and operating system more stable or reliable than another software grouping?

However, to do this, it would be necessary to set up and test each individual group of software components in an independent, controlled environment; this is an activity which is outside the scope of this research.

Similarly, were it feasible, it could be interesting to examine whether one subsystem reacted in a stable manner with respect to another subsystem. For instance, it would be interesting to examine the acceptability of mixing one supplier’s system of software sub-components with another system.

Finally, it could be interesting to make comparisons between like products as offered by suppliers of these systems. Unlike the previous sets of tests, the results of this test would require comparison of two subsequent commercial suppliers using the same data. Whilst this could be interesting for the tendering party such as a county council, three year tendering cycles would make this difficult to achieve. It is worthy to note that due to the underlying complexities within the proprietary aspects of this system, it may be quite difficult to compare like-for-like.

In the next two layers, i.e. between the passive and navigation layers, it is felt useful to examine the primary support mechanisms in the system. For instance, is there variance or deviation in the underlying dialogues? What is the success rate for questions? Is a null response quicker than a response with data? Does the data within the question cause a variance in the response time? If the response time is consistent, do outliers occur frequently and what is the likely magnitude of these outliers. Answering these questions would help determine the need for processes that compensate for spurious “errors” which are bound to occur in this complex and interdependent system. Its need is independent of commercial factors and is useful to all with an interest in the system. No research exists between these two layers.

Yet another aspect which could be useful would be to determine the merits of whether the national gazetteer should indeed contain all points (stops) everywhere, or just a reference to them. Furthermore, it
is useful to know the relative effect of whether the national gazetteer is held locally or accessed from a central source each time.

<table>
<thead>
<tr>
<th>Layer</th>
<th>Layer</th>
<th>Expected Response Time</th>
<th>What to Vary</th>
<th>Why Measure this</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active Navigation Internet Server</td>
<td>National Gazetteer Data</td>
<td>about 0.5 sec</td>
<td>Vary whether the NG is accessed from a central location or copied to the location of the ANIS</td>
<td>If the service is centralised, and it fails, then the whole country effectively disappears; on the other hand replication introduces administrative overheads and personnel costs</td>
</tr>
</tbody>
</table>

Table 20: Test to show merits of a local or centralised National Gazetteer

The final technical analysis assisting in a quantitative measurement of the system performance would be between the navigation layer and the software that sits on the “desktop”. This examination could look at additional uncertainty added by yet another layer to the system. Is the additional effect measurable? Is this a point that constrains the system, or is it further down? These analyses could support conclusions regarding where in the system the slowest link exists, isolated from the human element. It is worth repeating here that interaction between the system and the human is another area of research by itself, and with respect to this system, the "eight-second rule" (Keylabs, 2000) suggested earlier in Chapter Two is expected to be a constraining factor.

7.2.3 Logical Aspects of the System

Apart from the performance of the system technically, it is interesting to examine certain features within the navigation layer. It is here that the examinations for extensibility, and journeys with multiple modes and exchange points can be examined. However, these tests are data bound.

It would be expected intuitively that the system would always return an itinerary for every journey request. However, it is the case that the system does not always return an itinerary. Sometimes it is a technical fault. In other cases, it is lacking sufficient data to make a decision. It may also be that insufficient data exists from an element within the composite itinerary to compose a journey from the individual stages. It might be that no particular journeys meet the criteria specified. Thus it would be useful to determine whether the system should be returning data, and if not, why.

7.2.3.1 Parallel (Asynchronous), Serial (Synchronous), or Mixed

It was also stated that one of the objectives was for the system to give an answer to a cross boundary journey in a heterogeneous data and engine supplier environment. Should data be available, it could be interesting to investigate what sort of composite response times might be expected. However, more than one method exists to assemble this composite response time.
7.2.3.1 Parallel

In one case, it is possible to ask all questions in parallel. This means that the software continues to ask questions without waiting for the response of previous questions to necessarily return first. For instance, consider that the journey has multiple stages. In example, it has local journeys from the origin to the national exchange points, national journeys between two sets of exchange points, and local journeys at the destination side of the journey from second set of national exchange points to the final destination. In this case as shown in subsequent figures, many linkages exist. To determine the departure time for the middle set of journeys, one first must know the arrival time from the first stage of journeys (from the origin to the first set of national exchange points). However, if the journey navigator requests all the journeys between the origin and the national exchange points during the entire day, it need not wait for the results of these queries. It can send the requests for the second and third stages at the same time as the requests for the first stage journeys. In this manner, all the requests are parallel.

7.2.3.1.2 Serial

In the all-serial case, the software waits to ask the second question until it has received some response from the first, or a time limit has been exceeded. Similarly it waits to send the third request until after the second request has been sent. If the journey has multiple stages, with multiple exchange points, a series of three subsequent matrices could exist, resulting in 40 requests. In the all-serial request, each of these has to be requested and the response received before the next one can be requested.

7.2.3.1.3 Mixed Serial and Parallel

The third case is a mixture of the first (parallel) and the second (serial) cases. Returning to the three-stage journey case, imagine five individual requests occur in stage one. In the serial case, each of these has to be done in turn. In the mixed case, these can be done in parallel. Once this stage is complete and all five requests have returned with responses, the requesting software then knows the arrival times of the previous stage, and can then use these to make requests for journeys in the next stage. This set of journey requests can be done in parallel too. Arrival times are then available for the computation of the departure times for the journeys in the final stage, which can also be done in parallel. Thus the stages are done in serial, but the requests within a stage are done in parallel.

7.2.3.2 Potential Merits of Asynchronous and Synchronous scenarios

Having a sampling of response times is useful, but clearly it could be interesting to see the effect of methodologies which mixed the various cases. For instance, would it be beneficial for a navigator to ask for large number of individual journey alternatives spanning a large time band concurrent with requests for journeys that are downstream and dependent? Or is it quicker to ask a number of questions sequentially, and ask for shorter ranges and perhaps quicker response times from the individual legs, but having to accumulate time in the meanwhile? Perhaps a mixture of these produces a desirable effect.
It is perhaps obvious that whilst these comparisons would not be expected to have much effect on simple journeys, such as a journeys within a single database or a single mode journey, the usefulness of the comparisons would be expected to increase for more complex journeys types. Therefore, this experiment could have further saliency if different levels of complexity were introduced and compared.

For instance, image the first figure (22). It is representative of how a small set of matrixes might be connected together for an examination of a simple composite journey between one bus stop and five national stepping off points (three rail and two coach). These five origins on the national transport network are matched with four destinations, and subsequently four possible bus journeys to the final destination.

Should the designer of the navigator software desire the accuracy needed for a comprehensive system, which included the possibility of going from address to address, the small group of matrixes might have to be more complex. The following figure illustrates this graphically.

To explain, the first matrix from the left would represent a selection of five walking journey alternatives to the nearest five bus stops. The next matrix is from the five bus stops to the (nearest?) three rail stations and two (principal?) coach stations. These five stepping off points can be matched to four stepping off points at the destination. These have four bus stops which then connect to four walk links to the destination. This results in 60 linkages.

Now the first five could be asked in parallel, and then the next twenty-five, and then the next 10, followed by the next 16, and finally, the last four. Should all systems respond perfectly, and presuming a one second response time, these matrixes require at least six seconds to compute, should one include a second for splicing it all together. Whether this is realistic or in fact achievable is speculation at this stage.
Could it be done faster if all 60 linkages were done in parallel? Or would it be best to ask it in serial, with 60 sequential requests in total?

The concept is further relevant as at each of the inner points, current journey planners presume a simple walk link. As can be examined later as an issue with data accuracy, introducing walk link matrices can further complicate the individual exchanges between the inner matrixes. The importance for integrated ticketing becomes apparent as to queue at any point within the journey could cause serious distress for the traveller, and as a result cause the traveller to miss the connection. The result of this would be another matrix of enquiries at each stage on the system.

It can be seen therefore that—presuming good data were available—it could be interesting to determine if a variance in the composite response time results from changing the method of assembling the matrixes. Assuming such a variance exists, the derivative experiment of determining if a stable ratio of “cost”, i.e. increase in response time, can be formulated as a function of the journey complexity and the method of assembly is also interesting. It can be seen that matrix of possibilities could arise in this situation, and these are listing in the following table.

<table>
<thead>
<tr>
<th>Assembly</th>
<th>Journey Type</th>
<th>Response Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Serial</td>
<td>Simple Transport Network (Fig.20)</td>
<td>?</td>
</tr>
<tr>
<td>Parallel, Sequential</td>
<td>Simple Transport Network (Fig.20)</td>
<td>?</td>
</tr>
<tr>
<td>All Parallel</td>
<td>Simple Transport Network (Fig.20)</td>
<td>?</td>
</tr>
<tr>
<td>Serial</td>
<td>Transport Network with Walk trips</td>
<td>?</td>
</tr>
<tr>
<td>Parallel, Sequential</td>
<td>Transport Network with Walk trips</td>
<td>?</td>
</tr>
<tr>
<td>All Parallel</td>
<td>Transport Network with Walk trips</td>
<td>?</td>
</tr>
</tbody>
</table>

Table 21: Sample Tabulation Comparing effects of Assembly and Complexity on Response Time

It might be expected that the serial response time of the transport network with walk trips (i.e. Figure 22) case would have the worse response time, followed by the serial case with the simpler transport network only case, by a factor of 60 to 19, presuming a common engine time. However, whether the parallel sequential case will be faster than the all-parallel case is not clear. The reason for this is that the response time is expected to be a function of the necessary increase in data requested, and its cumulative impact on the composite response time, exclusive of the additional time the all parallel case would be expected to take to splice the resulting matrices.

However, it could also be a function of the line speed and capacity of the network between the various components of the system. Or, it could be a function of the interaction between the passive and active system, especially if the passive system frequently fails, and requires repetitive requests from the active system to compensate. This point emphasises the necessity of basic, fundamental research of the characteristics of the interaction between the passive and active layers, so as to enable the designer to remove a degree of uncertainty in the experiment caused by the lower layer.
7.2.3.3 Extensibility

With respect to extensibility, it is useful to know if there is a significant increase in time to determine a journey solution by the system if the geographic area of the system is increased. For instance, what happens to the response time of composite journeys between two points previously in the area of the system if additional members are added to the system? Were the system small and then increased significantly, say 100 fold, would that impact on response time of previous response times? This experiment is data bound too. However, it might be possible to simulate it.

Similarly, is it necessary for the individual members of the system to be relatively the same size, or would a variance in the system’s members be tolerated? For instance, if most of the databases and engines were representative of a rural county, with low frequency services and thus small databases and quick engines, would adding a large metropolitan, high frequency, slow engine area to the system affect the response time between two small rural systems? This question is particular relevant for England, as the density of services could vary perhaps 1000 fold between a large rural area and a relatively small urban area, if the Highlands were one example in comparison to London, Manchester or Birmingham.

7.2.4 Accuracy of the System

The system was designed to accommodate diversity in levels of data accuracy, so long as certain rules of consistency were obeyed. However, it is absolutely crucial to look at the accuracy of the data within the system, as the greatest efficiencies are useless from the perspective of the traveller—presuming the traveller is assumed to be demanding as some journalists would lead the population to believe—if the data is imprecise and inaccurate.

More to the point, it will be confusing to the traveller to be provided highly detailed information at one stage in the journey, and highly ambiguous information at a different stage of the journey. This situation is not critical if the traveller is familiar with the origin, and the origin stage is less detailed. However, should the destination stage be less detailed, and the traveller unfamiliar, this situation would be catastrophic to the traveller’s level of confidence in the information provided. As no known direct or indirect method of determining this need exists, and the cost of providing detailed information uniform everywhere, it is perhaps relevant to suggest that the highest level of detail possible be provided soon.

Setting the needs aside though, the relevant point for this research is not what is best for any particular party and why, but instead a measurement of what exists. As before, it could be interesting to quantify the benefit of better information and its cost on response time. However, this cost is decreasing as a function of hardware capability and improvement in methodologies. Such is the magnitude of the progression that the benefit will soon outstrip the cost, if it has not done so already. Alternative benefits could be gleaned from feeding the resultant data back into the system to assist in the planning of services.
However, such tangents—whilst valuable—are mere opinions and relative to the perspective of the party concerned. The state of the data is the matter at hand. How does one measure the state of the data though? One measure is to compare the term with the relative area it encompasses.

For instance, if a journey planner allows such ambiguity as to say get the traveller from Southampton to London, the ambiguity of this can be measured readily by comparing this question to some thing similar. To explain further, a table might be set up comparing the above question to get the traveller from Shirley in Southampton to St James Park in London. Further refinement would suggest a logical progression of get the traveller from the corner of Shirley Precinct in front of Iceland to the north end of Great Portland Street near the Royal Institute of Physics.

Were it not so obvious that London is a large area encompassing several hundred square kilometres, and Southampton somewhat less but still quite vague on the one hand, and the last example which references a few metres square on either end, then it is possible to suggest tabulating these differences. However, as it is obvious, it may not be necessary to rigorously argue the point.

It might be interesting however to see the shear magnitude of these differences in a quantitative manner. Furthermore, it could be interesting to compare this ambiguity as a function of a particular supplier of journey planners. In the defence of the journey planner suppliers, such ambiguity is sufficient if the presumption that a rail journey planner need not mention the fact that it is implicit that the ambiguous label of Southampton refers to the Central Rail station. However, it is not always explicit to the user that this is the case without some foreknowledge that the journey planner is provided by Railtrack, and Railtrack is the rail infrastructure provider and thus only concerned with rail stations.

In a market situation where bus companies run rail services and vice versa, this defence is becoming increasingly weak. The reason for this is that the number of bus stops included in the rail station list increases as a result of some rail services being replaced by dedicated bus links run by the train operating company. The inverse situation where a bus company runs rail services is not unheard of either. It is thus well illustrated the paramount importance of precise and accurately labelled information describing the name of a particular "place".

7.2.5 Summary

As the processes developed during the research for this thesis had not been previously possible, the primary focus of the data and analysis is to look at repetition and consistency. Can the experiments be repeated? Are the results different? Do similar tests produce similar results? Are there outliers? Is the system reliable or stable in a technical sense?
Whilst the research is in a sense data bound as very little data exists from which to do complicated analyses on journey engines from different suppliers and their relative effects, and resources are limited to NT systems and software, much can still be investigated. To review, it is interesting to indicate the possible effects of decisions made on the architecture, the messaging, and the splitting of the geographic naming hierarchy and the associated databases. It is also interesting to predict the potential benefit of structuring the requests in a parallel, mixed, or serial manner. Thus the research concentrated on testing, results, and contributions to answering these questions.

7.3 Performance Measurements of System Sub Components

The research in this thesis sought to produce a series of tests that would demonstrate reasonably well that the concepts worked in a technical sense, even if the accuracy of response in a real sense and its usefulness might be in dispute. In order to do this, fundamental research experimented primarily with the individual dialogues necessary to "prove" the system as a whole, and less so on the composite results of the individual dialogues.

7.3.1 Influence of Software within the Passive Layer

As suggested in the previous section (7.2), the skills necessary to examine the various configurations require an in-depth knowledge of three separate operating systems, databases in use on those operating systems, and in addition, the interaction of network protocols between the operation systems and the network. These fall broadly into IBM mainframe, UNIX, and Windows. Macintosh systems can offer a similar functionality but are not widely used in this area. The key point for the research described here is that insufficient resources existed in all four areas. Thus the research in this thesis was limited to testing Windows-based operating systems.

Within the Windows operating system, it is suggested here that it could be useful to test two different groupings of software programs. The first group is a CGI program written in Visual Basic which accessed a database (Access 97), on a Windows NT server, Service Pack 4. The second group was using what is called an “Active Server Page”, which is just another method of implementing CGI. It also was connected to a database (Access 97). Later research could investigate the implementation of SQL Server and NT 2000.

Whilst Franklin was quoted in Chapter 6 that the Unix grouping would be considerably faster, it was also stated that Microsoft itself suggests that the desired grouping is with the Active Server Page. They profess that the reason for this is that the Active Server Page is an in-line-process that shares the same memory and thread as the server. It is not know to what extent this is true, as no experiments were specifically carried out here regarding these layers in isolation. The reason for this is that all of the software groupings used in this research are proprietary. However, later tests at high levels in the system
suggest that it is not relevant whether Microsoft Internet Information Server is used or not, nor whether the CGI or ASP is used, so long as a Visual Basic “Form” is not used.

7.3.2 Performance of Components below the Active Layer

This section examines in detail the characteristics of the following three calls between the active and passive systems:

- calls for individual places given the name of a particular "area"
- calls for relevant exchange points given an individual "place"
- calls for relevant journeys between an origin and a destination within a database area

One analysis sought to determine the reliability of getting a response once the question had been asked, as it might be intuitive that if another layer is added, with seven sub-layers, it could have a significant impact on reliability. So for instance, if experiment sent several hundred questions, did a response come back each time? Within these experiments it is possible to introduce some variance.

For instance, if the question asks for all the places within a large conurbation, the response would be expected to be quite large with respect to a similar question requesting the places within a small village. However, would the response time vary significantly? Another question of interest is if no journeys are available, does the system respond quite quickly to say that no journeys are available, or is it quicker to get journeys back? Similarly, are complicated journeys (i.e. many trips), more time consuming than direct journeys (i.e. only one trip)? Therefore this section seeks to display and analyse the answers to these sorts of questions.

7.3.2.1 First Test Apparatus - Early Trials

As an initial set of tests, a software program was set up that repeatedly asked the question what places are in this village or locality. The test data had 508 villages in it, i.e. all the known localities in two selected counties in England at the time of the test. The test environment used a specially written program using the Visual Basic programming language to fire the questions sequentially at two identical software engines programmed by a data supplier. These data engines were connected to two exposed data sets containing places and the related villages, one for county A and one for county B. By "exposed" it is meant that the data was available for viewing and editing. The software engine interrogating the data was pre-compiled however, and its algorithms are not known; though it is known that the software engine included a "form" object. This state of being precompiled meant that it was difficult to discern just what was happening in great enough detail to indicate something useful from the results of the tests.

Furthermore, these tests were conducted in the summer of 1998 at a stage where Microsoft was just beginning to adjust to the fast pace of developments in the field. Service Pack Two had just been released and this had reduced the time to respond from two minutes to something less than a few seconds.
However, by the summer of 1999, Microsoft had released a more stable method of access to the internet with its Internet Information Server. Initial tests with this technology showed an order of magnitude improvement in response times, and also suggested that response times from a Microsoft based system would now be similar in order of magnitude to the UNIX based systems. As a result of the tests and the introduction of reputedly more reliable, faster methods, the supplier of the designed software decided to change its base architecture and software too.

Therefore, rather than rerun the tests under the old arrangement for greater rigour, it was decided to set up and program a completely independent software and test environment under the new methodologies, and base any benchmarks on these new methods.

7.3.2.2 Second Test Apparatus

As the above results showed the stated inconsistencies, a more comprehensive test apparatus was set up in the summer of 2000. Two identical databases were created and set up on two Pentium class machines with 128 MB, IDE hard drives and NT4 service pack 4, IIS 4, VB6 with service pack 3. Requests were set up so that the measurements could be intersected at more points in the process, and thus provide a more precise indication of what exactly is happening. As before, the research sought to benchmark performance of the three types of requests within the system. The following figure illustrates the apparatus and the test points.

![Test Apparatus Diagram]

**Figure 24: Test Apparatus**

The figure shows that an internet "form" using the POST method served as the "control" or management software. It has options for the number of tests (1 to 2000), the values to be sent (text box entry), and which test to perform (place, exchange, or search).

These are then sent to the simulation script. It processes the options and then loops through the various software code until the test has completed the requested number of times in the requested fashion. It also records the results to the database, and outputs the results of each test to the control browser as it completes each of the test runs.
The swap script is a short piece of software which swaps the test between one machine and the other. It does this by writing an integer (0 or 1) to a small file on a physical hard disk. If the value is naught, then the swap script reads the input string, (timing this action also), and sends the test request to the first machine. If the value is one, then the swap script reads and sends the input string to the other machine.

In either machine, the CGI script reads the input string. It then performs the necessary action, be it place or exchange or search. If the action is place or exchange, it directly accesses the database, executes the command to run the query on the table data. It then parses the data, composes a string response, and sends this text string back through the test system.

If it is a search query, the CGI script must read the input text string, translate the values inside to that of the subservient search engine, and then it sends the request via HTTP to the remote journey search engine. The remote engine resides on the same machine, so it is sending the request to itself. The 'remote' engine then processes the request and sends a response back to the CGI script. It sends the input text string from the remote search engine back up the system, recording additional timing details along the way as it did on the way down.

It is this timing data that is important in the tests. It can then be used to provide a number of measurements at different points in this complex process. If analysed and assembled in a particular manner, these measurements may provide an indication of the relative merits of various decisions made in this research, and provide a means to hypothesis how future systems might be expected to perform given a particular construction.

7.3.2.3 Test Conditions

It is useful to further detail a number of test conditions before going on the tests and measurements.

- One request after another; each request was completed before the next request was started and no other tests were run at the same time

This test condition is important from the perspective that it isolated the response time from the effects of multiple requests coming into the system at the same time. Whilst it is interesting to know the effects of loading, there exists commercial research into this and thus it is not detailed here.

- No network traffic existed except for the interruption of global broadcast messages from the rest of the wide area network, as all network connections were via a 10/100 auto-sensing switch

This test condition was necessary to isolate the effect of other network traffic on the tests. This had been a problem in previous trials in 1998, and unexplained patterns resulted.

- Values are representative of a Windows NT environment with all hardware running on IDE drives and 128 MB RAM;

This test condition further limits the work to that of an NT environment. Whilst research using another operating system and internet server software could be used, these resources were not available. The effect of the IDE hard drive would be to slow the system performance down of any parts of the system
which access data on the disk (i.e. the search engine or the writing of the swap file), with respect to parts of the system which only use memory. Another reason IDE was used is that they are less expensive and easier to configure and install than SCSI drive. The penalty in performance between using a SCSI and an IDE drive was not measured.

- Network cards are all 100MB except for the machine running the swap script and simulation script (both on same machine at 10 MB), and control browser on same machine.

The last test condition simulated a "high speed" data interconnection. In practice, the industry may not be able to afford such a "high-speed" connection between individual search server services. A lower network speed could have the effect of increasing the network time, especially if the text string packets are large in size. If the text packets are small, then the relative effect of network connections with less capacity would be expected to be smaller. It might be expected to increase the relative proportion of the response time taken up by the network communication.

The database file type in these samples was Access 97. This database software is known to have a limitation of about 200 simultaneous users. This limit was not affected as only one user accessed the software at any given moment. A copy of the same database was used in the trials on both machines, and the scripts were identical except for code stating which machine it was. Visual Basic Enterprise 6 service pack 3 was installed on both machines and the machines ran under an NT 4 windows operating system with service pack 4.

7.3.2.4 Measurement & Accuracy

The stated accuracy of the measurement of time was set a 1/1000 of a second or one millisecond by Microsoft. In setting up the apparatus described, it was discovered after running numerous tests that the same numbers were being returned each time. It is highly suspicious to get exactly the same number back, albeit at an irregular pattern, to a significance of eight digits. Upon investigation, the reputed accuracy was determined to be in dispute, as all of the measurements matched an exact multiple of 1/256. This suggests that the accuracy is indeed every four milliseconds instead of $1/1000^6$. Whilst it is important that this accuracy placed a limitation on the granularity of possible measurements, the processes being measured are sufficiently complex so as to allow detection even at this coarse interval.

7.3.2.5 Required Number of Measurements

A series of tests were run to test the system for reasonable measurements and patterns. Once these initial tests were created and run, it was felt useful to determine how many iterations would be necessary for an established pattern to evolve. Whilst at least one reference (Figliola, 1991, p 133) gives a formula to suggest a method to determine how many measurements are needed, it was decided to increase the number of iterations by an order of magnitude until a stable histogram was obtained.
Table 22: Histograms at different measurement points for a PIII-500 with 50 iterations

In the first test, 10 repetitions were used; five for each machine. The histogram for this test did not show any useful pattern and was felt to be too small. The test was repeated by a second test with 100 repetitions. This test then had 50 iterations for each of the two machines. Whilst a pattern emerged, it was not clear by comparison of the first test with the second that the results were consistent. Thus a third test was completed with 1000 iterations; 500 for each machine.

Table 23: Histograms at different measurement points for a PIII-500 with 500 iterations

A definite pattern can be seen by comparing the histograms for the 50 iteration sample with the histograms for the 500 iteration sample on the Pentium III - 500 MHz machine. This similarity of profiles suggests that 50 iterations of the same test are sufficient to ensure a consistent measurement. Therefore, in later testing, only fifty tests per machine would be necessary for each trial. However, for the place and exchange trials, the response times were sufficiently short to permit samples of 500 iterations to be conducted without undue hindrance.

It is remarkable however that a definitive "hole" exists in the histogram in the centre of an otherwise "normal" distribution. Looking at the "Time in Remote Script" and "Time in Remote Database" shows that in between the values of 0.0507813 and 0.0585938 lies one value in the middle. It is felt that this is
due to the limitation found in the quoted accuracy of the timing measurement and measurable accuracy obtained in practice. With larger intervals, this "hole" would not be noticeable. As such, it is important to note that these measurements are can only provide an indication of future performance to within a certain degree of accuracy. This research limits the accuracy at 1/128 of a second, resulting from this "hole" in the histogram at 1/256 of second intervals.

In the "Send To Remote Time" and "Total Time" columns, it can be seen that the pattern of values broadly mirror each other. It can also be seen that the measured "hole" has been reduced to one large peak, even though the interval of measurement remains the same in all columns. This pattern could then indicate that any variance in the total time is largely a function of the variance in the network time necessary to send and receive the text string message from one script to the other script. However, with the inaccuracy of the measurement, it is difficult to be conclusive.

7.3.3 Trials of the PLACE command

To review, three main issues are of interest. If the engine did not find any matching criteria, how long did it take to determine this? Secondly, if it did find a matching sample, how long would this be? If it found a significant number of places matching the criteria, did this take significantly longer? Was the significance due to network overheads? Thus, using a database approximately one megabyte in size representing only one county, three tests to discern the answers to these questions were run.

In answering the first question, a known value was sent to the search engine that was well out side of the geographic area relating to the database in question. To answer the second question, a small village was chosen which was known to have six (6) bus stops within the village boundary and thus associated with the village name. No other village was near by, so the bus stops were not related to any other village either. The third trial tested a "large" town, which was known to have approximately 100 bus stops (95 in exact number) associated with its place name. This number is about an order of magnitude greater.

In making these measurements, the network time and the request read and file swap times were recorded. These additional measurements can provide an indication of the relative magnitudes of the various actions occurring in this series of tests. Finally, the same tests were carried out on two different machines so as to ensure that a similar operating system and system configuration did not produce incomparable results.
The previous figure summarises the results of 3000 test results, 500 hundred for each case on each machine. As expected, the null response was the fastest on average with the smaller request of "Winslow" returning a response time slightly longer with its six bus stops. The noticeable difference is with the larger request for Aylesbury. This response time is on average 50% larger to produce a response time of just under a tenth of a second. The faster processor machine is full hundredth of a second quicker in this case.

This large difference (50%) over the smaller village is significant, as it suggests that breaking the hierarchy at the "community" level may be a good decision. Relating this back to the numbers provided in section 5.3.2, the reason for this is that a county is expected to have perhaps three or four thousand bus stops in it, and perhaps up to 250,000 potential departure points (i.e. 25 million addresses divided by 100 counties). Extrapolation might suggest that the response time for a 250,000 points might be too large to wait. Even if reducing to just 500 addresses per village and thus an estimated 0.2 second response time, it would require a great deal of scrolling by the traveller, a long time to read by the call centre operator, or some clever software to digest the list into manageable chunks. Therefore, it is suggested that the decision to encourage a flat hierarchy at the village level has significant merit from this perspective. Whether it can be made feasible through practical measures in actual implementation on a national scale may be another matter.
Finally, it is useful to review that for the "place" command, the values shown on the lower portion of the bars in Figure 26 measure the actual time spent in the database being searched by the engine. Literally, it is the time to execute the given query within the database. The network times, shown on the upper portion of the bars in Figure 26, in comparison averaged an addition 0.015 seconds in the short responses, and just under twice that in the longer response. In all cases however, the network time measured, on average, just over 20% of the total response time (see the left of Figure 26 for percentage of network and engine time as a function of total time).

7.3.4 Trials of the EXCHANGE command

In the testing of the exchange command, it was not felt useful to test for null responses again, as these would be similar to the test above. Similarly, it was not felt useful to test for a wide range of returned responses, as these too would be fairly static. Obviously the system could not cope with 100 possible exchange points as the resultant matrices could result in catastrophically long response times for the traveller. Thus this testing scenario concentrated on repeating the experiment to see if two independent trials of 1000 tests resulted in similar means and other relevant summary statistics.

As a result, the tables in this section are grouped so that results from the same PC are shown together. For instance, both data samples from the Pentium II - 450 are shown together. The same is true of the two results from the Pentium III - 500. Thus whilst Data Set One with the Pentium II - 450 was taken at the same time as Data Set One with the Pentium III - 500, they are NOT shown together.

133
It can be seen here that both data set one and data set two show broadly the same results over the sample size of 500 iterations. Given the stated level of inaccuracy measured earlier, this similarity of results suggests convincingly that the tests are repeatable, as expected. It could be noted that the individual deviations vary slightly but it is not felt to be anything other the effects of precision on the measurements.

Comparisons of these data values also show a high degree of consistency. Exceptions to this lie in the "IP Swap Time" and the "Read Message Time" values. Here the means are significantly different, varying by 50% and 300% respectively, suggesting that something could be concluded. However, as the mean response time in all cases is lower than the stated accuracy of the measurement, nothing can be concluded definitively without a more accurate means of measuring the phenomenon.
As with the place command trials, these values are small, even going through the added layers of the network to request the time. What is of significance is the difference between the "Send to Remote Script" and the "Time in Remote Script". This is approximately 0.015 seconds on average, and represents the additional delay in holding the information remotely. However, it is necessary to consider that this measurement was made under a "high" capacity network (10 Mbits/second) and over a very short distance (a few metres). In a practical implementation of the system, it is supposed that the systems could be considerably farther apart. If this is the case, then the temporal effects of travelling that distance at the speed of light must be considered. Whilst small, they could be significant. Furthermore, if many requests are travelling down the line at once, this could have an impact on the true performance also. Both of these need to be considered should a real implementation of the system eventually take place in an operational sense.

Even so, should the leased line be a reasonable capacity, it is felt as a result of the above measurements that the decision to keep the exchange points at the local database and not collect them centrally and redistribute them has not added any appreciable detriment to the overall performance. It is broadly felt the benefits of higher accuracy and less data maintenance are more important than the small temporal penalties of keeping the data close to its source. However, these indicators would have to be examined with respect to the quality of line that the system is connected to the internet and the cost of providing that particular line quality in a practical implementation of the system.

7.3.5 Trials of the SEARCH command

In testing the search command, several issues are relevant. One of the issues is to determine an expected response time. Within this response time is the desire to determine the individual components of the response time and their absolute magnitude and relative size. This data will then provide an indication of how navigation software could be designed, and how the overall system might be implemented. It can also act as a guide in determining individual areas to investigate should a particular implementation scenario prove inefficient. Integral to these examinations is the relative merit of whether it is best to ask all the questions for a particular origin and destination pair at once, or instead, ask a sequence of questions in a particular manner. This last issue is of particular importance to call centre operations which are time critical.

Therefore, the following analysis is separated into two parts. First, the performance was examined by asking a select group of questions using a "short" interval. Essentially, the question of the search engine is to provide the next X journeys that satisfy a given time criteria (arrive before 16:00 on this date), where X is a small number such as two (2) or three (3). Subsequently, this study looks at the other side of the issue, asking the search engine to return all journeys within the next Y hours, where Y is effectively a whole day (20 to 24 hours). This request would normally return many journeys and a very long packet of characters. Thus the internet overhead would be expected to be measurably larger. The order of
magnitude of the search time to the engine time will be of considerable interest. Finally, it will be interesting also to look at the relative impact of reading a request string and writing to a file, and then re-sending the request. These actions are of interest as it may be necessary to marshal requests to one or more search engines, should the search engine or network time be significant. Thus the relative magnitude of this marshalling action is determined also.

Related to this is the concept of extensibility, as these actions will provide an indication of the relative effect the chosen, distributed design has overall. To review, if these marshalling actions take a long time, then the extensibility of the model in a global system will be hampered. Therefore, it is the relative magnitude of the effect of different configurations of these technologies to the issues of journey planning that this research seeks to provide a clear indicator.

7.3.5.1 Sequence = small

Having determined that the histogram does not change significantly above a sample of 50 iterations, ten origin and destination pairs were chosen. These ten pairs sought to exemplify the different characteristics of the engine and the data. It was suggested by the designer of the engine that the response time would increase as the distance between the origin and destination along the network increases. To clarify, whilst the absolute geographic distance between Swansea and Bangor is a few hundred kilometres, the distance along the transportation network is much longer, as no rail lines exist to connect these places directly. As the route has to travel via Cardiff and Chester, the distance along the network is actually much longer, perhaps two or three times the geographic distance.

It is further hypothesised that if numerous possibilities exist between an origin and a destination, then the response time of the engine is expected to increase also, as a result of having more options to rule out. Whilst this is a similar issue to that of the distance along the network, it is expected that an origin and destination pair with many options and a long distance would take longer than a journey with a similar distance along the network, yet fewer options. An example of this would be from the West county to the North, in comparison with from the South to the North of the country (Scotland). Bearing these issues in mind, this research choose the following origin and destination pairs in testing the performance of the engines in extracting the journeys with the shortest journey time from the scheduling database.
Table 28: Origin Destination Pairs for Search Request Tests

The first origin and destination pair are relatively near. The second pair takes the traveller from the north of Scotland to the north west corner of Wales in a sort of "S" shaped trajectory. The third pair follows the same trajectory, but in an elongated fashion to the south west of England. Origin and destination pair four are two adjacent stations. The fifth pair is an reverse "C" pattern, going from the south west of England to the south west of Wales. The same is true of pair six except it is from the south west of Wales to the north west of Wales. In this case too the absolute distance is much shorter than the distance along the rail network. The seventh pair is the second longest origin destination pair stretching from Southampton in the far South to Aberdeen in the north east of Scotland. Pairs eight and ten are across the country from West to East, and from Wales and south west England to the south east of England respectively. The ninth pair is from south east of England to the north east of Scotland. Absolute distances are distorted as the actually distance would travel via an arc between the origin and destination, and not via the computed value of a straight line on a flat surface. So whilst the distortion is small in the shorter distances, in the longer distances, the actual distance travelled would be longer. However, the computed distance is only meant to be a rough approximation.

In conducting the test, a batch file was set up to measure the time at each particular point in the process, as diagrammed previously in Figure 24. The response time shown here corresponds to points 8 and 9 for the Pentium II - 450 and 14 and 15 for the Pentium III - 500. As stated previously, 50 samples were taken for each origin and destination pair. These are tabulated below.

Table 29: Measured time for the engine to respond to enquiry request
Unlike the previous two tests, (place and exchange), the engine in this case is proprietary. As such it was necessary to make an Internet call to the engine. Thus, the values above include the Internet overhead. By subtracting the mean Internet overhead as measured between the nodes on either side of links 7 and 10 for the PII and 13 and 16 for the PIII, it is possible to estimate the actual engine time. These estimations are tabulated in the following table.

<table>
<thead>
<tr>
<th>Test Index</th>
<th>Railway Station</th>
<th>Pentium II - 450</th>
<th>Pentium III - 500</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Aberdeen</td>
<td>0.287891</td>
<td>0.241641</td>
</tr>
<tr>
<td>2</td>
<td>Aberdeen</td>
<td>0.497422</td>
<td>0.415547</td>
</tr>
<tr>
<td>3</td>
<td>Aberdeen</td>
<td>0.557813</td>
<td>0.463369</td>
</tr>
<tr>
<td>4</td>
<td>Exeter St Davids</td>
<td>0.1175</td>
<td>0.087767</td>
</tr>
<tr>
<td>5</td>
<td>Exeter Central</td>
<td>0.379297</td>
<td>0.311406</td>
</tr>
<tr>
<td>6</td>
<td>Swansea</td>
<td>0.573281</td>
<td>0.478516</td>
</tr>
<tr>
<td>7</td>
<td>Southampton</td>
<td>0.734688</td>
<td>0.539653</td>
</tr>
<tr>
<td>8</td>
<td>Bangor</td>
<td>0.490313</td>
<td>0.404844</td>
</tr>
<tr>
<td>9</td>
<td>Rainham</td>
<td>0.651406</td>
<td>0.528672</td>
</tr>
<tr>
<td>10</td>
<td>Exeter Central</td>
<td>0.375703</td>
<td>0.247031</td>
</tr>
</tbody>
</table>

Table 30: Estimated response time of the routing engine

It can be seen from the table that the response times more closely match the expected time, unlike the tests run in 1998. It can also be seen that the times vary somewhat between a tenth of a second to eight tenths of a second. Roughly speaking, the longer the distance along the network, the greater the time taken to respond to the enquiry. It is not a linear function however, as the engine will always take some time to navigate certain overheads associated with opening and closing database handles and so forth. Likewise, the longer times will tend to group towards the eight-tenths measurement. It is important to note that the null response (no journeys but a successful query) will reside near the lower end of the measurements. This is not explicitly shown here but numerous tests demonstrated it.

Therefore, as a rough guide, the response times did increase with an increase in distance and complexity. For instance, the Southampton to Aberdeen pair did take longer to respond than the Exeter to Aberdeen pair, and these two took the longest of those origin destination pairs sampled. The reason for this was that relatively few route options exist to go between Exeter and Aberdeen whilst a fair number of competitive options via different routes (Oxford or London) are feasible between Southampton and Aberdeen. It is useful to note that in these test pairs, the time to journey between the origin and destination and the distance to journey between the origin and the destination are presumed to have an approximately linear relationship. If however this research were to compare a route such as Birmingham to London, where two routes of similar distance exist with a significant difference in journey time, it could be that the engine responds quicker to the route with a shorter journey time (i.e. not Princes Risborough). Due to the commercial nature of the algorithms underneath, this question is purposely left unanswered in this work.
A question of considerable debate in some circles was whether the response time of the engine was of the same order of magnitude as the network time. The following chart depicts the relative portions of time as a percent of the total time for each of the ten pairs of points.

Figure 27: Relative Engine and Network Mean Time of 50 samples on a Pentium III - 500 MHz

This figure shows clearly that the relative network time is roughly stable at 0.018 seconds and on average, takes less than ten percent of the combined engine and network time. As a guide then, the engine time in this benchmarking took ten times the time of the network time. No relationship exists between the network time and the engine time however, as they are completely independent.

A comparison of the fourth and first origin and destination pair lends weight to this by illustration, as although the relative time of the network is increased to twenty percent in the fourth pair, the absolute time is still around the stable time of 0.018 seconds. In origin and destination pair one however (fourth from the left), the time to compute the answer is longer but the mean network time smaller than in other origin and destination pairs. The same is true again as illustrated in origin and destination pair seven. Origin destination pair five illustrates that a longer network time can occur with a short engine time, thus suggesting that the inverse does not have a correlation either.

It is however necessary to note that sometimes the network time can be considerably longer than the engine time. Whilst this occurrence is infrequent (once in 1000 measurements), the resultant bias is considerable, as shown in origin destination pair ten. In this case, one network time measurement at nearly 3.5 seconds was recorded. The reason for this occurrence is not known, but could be a result of the manner in which the underlying network communication is completed. Normally in the "handshaking" which goes on underneath in the socket communication, a network card (A) makes election on to the network (cables, optics, or whatever). Should the network be carrying a packet at the exact moment and place that the election is made, then the network communicates to the network card that it is "busy".
network card waits a random interval and tries again. It is unusual for it to be busy twice, but possible. Similarly, the network card on the other end of the conversation must go through a similar process. It is possible that both cards were busy in this particular instance, albeit rare.

In summary, the network time averaged less than a twentieth of the combined network and engine time, or the engine time is an order of magnitude greater than the network time. The engine time averaged at about 0.4 seconds and the network time averaged about 0.02 seconds. The frequency of large network times is less than 1 in 1000.

7.3.5.2 Trials of the SEARCH command (interval = large)

Having determined the characteristics of a search engine when asking for a small number journeys, (i.e. return the first journey, which departs after a given time), it was interesting to determine a sample of response times should the traveller require a selection of journeys. For instance, the call centre operator may be requested to give the traveller the departure time of the next six (or any other number) journeys so as to find one which matches some other criteria of particular interest to the traveller. This type of request could be serviced by subsequent requests for further information, or it could all be requested at once according to design of the call centre software. Thus this research repeated the above tests but with a change in the search command to ask for all journeys during a 20 hour period of the same day (effectively all journeys from 2 AM to 10 PM). The results from these tests are tabulated below.

<table>
<thead>
<tr>
<th>Test Index</th>
<th>Engine Time</th>
<th>Railway Station</th>
<th>Pentium II - 450</th>
<th>Pentium III - 500</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Origin</td>
<td>Destination</td>
<td>Mean</td>
<td>Min</td>
</tr>
<tr>
<td>1</td>
<td>Aberdeen</td>
<td>Aberdour</td>
<td>1.1848</td>
<td>1.1523</td>
</tr>
<tr>
<td>2</td>
<td>Aberdeen</td>
<td>Bangor</td>
<td>2.5723</td>
<td>2.5430</td>
</tr>
<tr>
<td>3</td>
<td>Aberdeen</td>
<td>Exeter Central</td>
<td>2.3909</td>
<td>2.3711</td>
</tr>
<tr>
<td>4</td>
<td>Exeter St Davids</td>
<td>Exeter Central</td>
<td>0.6929</td>
<td>0.6797</td>
</tr>
<tr>
<td>5</td>
<td>Exeter Central</td>
<td>Swansea</td>
<td>1.6700</td>
<td>1.6523</td>
</tr>
<tr>
<td>6</td>
<td>Swansea</td>
<td>Bangor</td>
<td>2.1115</td>
<td>2.0936</td>
</tr>
<tr>
<td>7</td>
<td>Southampton Central</td>
<td>Aberdeen</td>
<td>2.8009</td>
<td>2.7852</td>
</tr>
<tr>
<td>8</td>
<td>Bangor</td>
<td>Rainham</td>
<td>2.3116</td>
<td>2.3008</td>
</tr>
<tr>
<td>9</td>
<td>Rainham</td>
<td>Aberdeen</td>
<td>2.5271</td>
<td>2.5039</td>
</tr>
<tr>
<td>10</td>
<td>Exeter Central</td>
<td>Rainham</td>
<td>1.9226</td>
<td>1.9023</td>
</tr>
</tbody>
</table>

Table 31: Engine and Network response time for "interval" equals 20 hours

Again in this sample, one adversely large network time was measured. As the response times are longer, the ranges measured appear much tighter, though they still follow the same patterns as before. However, whilst this is consistent, the estimated network times are approximately twice as much as in the previous trial. This measured effect is the result of a much larger range of journeys being returned, which results in a larger string size, and thus more packets being sent back to the parent script. Thus the salient observation is that the network time effectively doubled by including all the options for a whole day.

Irrespective of this important observation, the time necessary by the engine was much larger. Comparing the interval with the sequence measurements taken earlier shows this.
Table 32: Approximate ratio comparing the time to retrieve a whole day's journeys verses just the next few

As the above table shows, by asking the engine to return the whole day's journey, the response time was on average (mean) four to five times as long. The above results also suggest that these ratios are independent of distance.

7.3.5.3 Relative merits of making a small sequence enquiry vs. a large interval enquiry

It is possible to indicate the possible effects of these results for call centre operations, presuming at the onset that computing resources are not a constraining factor as calculated in Chapter Three, section 3.1. If the traveller is requesting a simple journey (one or two stages), it could be suggested that from the perspective of the traveller, it could be quicker to make a "mixed" enquiry, i.e. each stage in serial and requests within a stage in parallel. If the majority of requests are for short distance or single mode journeys, then it could make sense to design the navigator to make "mixed enquiries". This could result in a shorter response time for the traveller.

Even if a repeat request is necessary, so to find a later departure, the time of two requests would still be shorter on this basis than requesting a whole day's journeys and making the requests within a stage and in additional all the stages in parallel. Normally the next two or three alternatives can be handled within a short sequence request also. In this manner, the traveller can have the next two or three journey itineraries quite quickly without waiting for a whole day to be calculated.

This design could have benefits and disadvantages to the call centre operator also. If the call centre staff are being requested take a short amount time to answer the call, the call centre operator would not like to have the staff waiting for the remote computer systems to respond. On the other hand, the staff may well appreciate the short pause to relax for a moment.
In the case of a longer journey with three or more stages to its enquiry, it could prove beneficial to design the software so that it asks all the requests in all the stages at once. In this manner, the call centre staff would benefit from being able to offer a greater variety of alternatives in journey departure/arrival times. If this type of journey were done in a serial format where only the journey requests within a stage were done in parallel, any requests for later or earlier departure information would require a whole new request at an associated cost of three or four additional seconds. This is because the sum of the times to request a three-stage journey would be nearly the same or only slightly less than the composite time of a whole day's request.

If computing resources are limited, then this may impact the system in two ways. If a particular network has only one or two passive servers supplying it, then once capacity is reached, other requests will have to queue. This has particular importance 1) if the journey has many stages, and 2) the journey has many possible exchange points at both the origin and destination relative to that network. Imagine for instance that the journey has three stages, five exchange points on the rail network at the origin, six exchange points at the destination, and only two remote journey planner servers for rail, with an average response time of one second for a whole day's journeys. In this case, the 30 requests would have to be queued in pairs, resulting in 15 seconds just to complete the second stage. Similarly, if the volume of requests were large at any one moment in time, then it would have a similar effect of queuing requests. The effect of load sharing is discussed later however.

It is also necessary to bear in mind that the above analyses reflect the condition that all engines perform equally. It may be the case the some engines, whilst sharing similar characteristics as discussed above, are able to improve on the magnitude of the measurement. In the short time this research has been done, the relative strength of the CPU has changed several orders of magnitude. The relative impact of this is that larger databases can be search in comparable times, or alternatively, smaller databases can be searched in less time. It is suggested for instance with particular relevance here that some of the passive systems will have higher speed equipment than others. This is particularly important in the "national" network systems such as the rail and coach systems.

For example, if the three or more stage journey in terms of the networks being searched contained a journey that was far apart in one stage, it could prove disadvantageous to request all stages in parallel, as this particular stage could unduly bias the overall response time. For instance, if stages one and three were small networks and resulted in a short time, and stage two was a small network but entailed several requests between Scotland and South England on the rail network, then this longer response time would bias the overall time towards three (3) seconds for a whole day's journey. Asking for a small sequence of journeys in the second stage could present the opportunity to reduce this to nearer one (1) second whilst still providing the flexibility of two or three alternative departure/arrival times for the traveller. Asking for the second stage first could reduce the number of exchange points for journeys out of, in to, or between metropolitan areas with many main termini stations. It is worth repeating that these response
times are reflective of a small schedule database (less than 10 MB). Larger databases could expect much longer response times.

It is also necessary to state the ratio of interval to sequence time could decrease or increase depending on the algorithms and parameters used in the routing engine. The engine measured was a prototype and thus commercial versions of the engine could exhibit different behaviour depending upon the desired application. The relative impact of a shorter ratio would be to increase the viability of obtaining a “whole day’s journeys” for simpler (i.e. two stage, or perhaps even single stage) journeys. It could also improve the response time should someone want a schedule for any particular service for the entire day.

In summary, the ratio of response times by requesting a whole day's journey verses just a small sequence (the next two or three) is four to five times as large. The additional network penalty (on a 10 MB/sec line) is approximately twice as large. If the line speed of the network were considerably slower, the impact could be considerably larger. Thus in consideration of using the interval command to provide a larger complement of alternative departures for journey itineraries, it is necessary to bear in mind geographical distance within the transport network, network line speed, and whether the journey is a long distance or mid distance class journey.

7.3.6 Extensibility, Load Sharing, and Farming

One of the objectives of this research was to look at extensibility. The above analyses suggest that, with respect to a global system, if the individual components are small, then it is possible to marshal the request to the appropriate resource. This statement arises from the above results which suggest that apart from the odd occasion, the relative time for a network connection is small in comparison with searching a database with an engine, whether it is for looking up information or determining a route. This suggests that marshalling the data is an effective way to accomplish the objectives of extensibility, load sharing, or as it is sometimes called, farming.

Extensibility can be accomplished by first accessing a database and then determining which of the subsequent data sources needs to be accessed. Load sharing and farming, whilst created for a different purpose, can be achieved through a similar mechanism to the implementation of extensibility. In the first case of extensibility, the process is to access a local gazetteer of information and determine the relevant internet address. In the case of load sharing, the process is to write to a temporary file and swap itself between one or more server machines in a “farm”.

The first process of extensibility is analogous to the either the place command or the exchange command, but higher up in the system. They showed a minimal look up time, and even when combined with an internet network time, still registered a measurement under a tenth of a second. This suggests that the penalty for a central, remote data source would be small temporally. Thus the majority of the risk lies with capacity of the central service, and whether the central service would be available 100% of the time.
The second process of load sharing was modelled by writing a value to a small temporary file and reading this file. The time taken to do these tasks was recorded and is displayed in the following table.

<table>
<thead>
<tr>
<th>Swap Server</th>
<th>Railway Station</th>
<th>Pentium II - 450</th>
<th>Pentium III - 500</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test Index</td>
<td>Origin</td>
<td>Destination</td>
<td>Mean</td>
</tr>
<tr>
<td>1</td>
<td>Aberdeen</td>
<td>Aberdour</td>
<td>0.000625</td>
</tr>
<tr>
<td>2</td>
<td>Aberdeen</td>
<td>Bangor</td>
<td>0.000234</td>
</tr>
<tr>
<td>3</td>
<td>Aberdeen</td>
<td>Exeter Central</td>
<td>0.000391</td>
</tr>
<tr>
<td>4</td>
<td>Exeter St Davids</td>
<td>Exeter Central</td>
<td>0.000313</td>
</tr>
<tr>
<td>5</td>
<td>Exeter Central</td>
<td>Swansea</td>
<td>0.000938</td>
</tr>
<tr>
<td>6</td>
<td>Swansea</td>
<td>Bangor</td>
<td>0.000391</td>
</tr>
<tr>
<td>7</td>
<td>Southampton Central</td>
<td>Aberdeen</td>
<td>0.000234</td>
</tr>
<tr>
<td>8</td>
<td>Bangor</td>
<td>Rainham</td>
<td>0.000469</td>
</tr>
<tr>
<td>9</td>
<td>Rainham</td>
<td>Aberdeen</td>
<td>0.000703</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Read Time</th>
<th>Railway Station</th>
<th>Pentium II - 450</th>
<th>Pentium III - 500</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test Index</td>
<td>Origin</td>
<td>Destination</td>
<td>Mean</td>
</tr>
<tr>
<td>1</td>
<td>Aberdeen</td>
<td>Aberdour</td>
<td>0.001484</td>
</tr>
<tr>
<td>2</td>
<td>Aberdeen</td>
<td>Bangor</td>
<td>0.001406</td>
</tr>
<tr>
<td>3</td>
<td>Aberdeen</td>
<td>Exeter Central</td>
<td>0.002031</td>
</tr>
<tr>
<td>4</td>
<td>Exeter St Davids</td>
<td>Exeter Central</td>
<td>0.001875</td>
</tr>
<tr>
<td>5</td>
<td>Exeter Central</td>
<td>Swansea</td>
<td>0.001563</td>
</tr>
<tr>
<td>6</td>
<td>Swansea</td>
<td>Bangor</td>
<td>0.000938</td>
</tr>
<tr>
<td>7</td>
<td>Southampton Central</td>
<td>Aberdeen</td>
<td>0.001797</td>
</tr>
<tr>
<td>8</td>
<td>Bangor</td>
<td>Rainham</td>
<td>0.001016</td>
</tr>
<tr>
<td>9</td>
<td>Rainham</td>
<td>Aberdeen</td>
<td>0.000703</td>
</tr>
<tr>
<td>10</td>
<td>Exeter Central</td>
<td>Rainham</td>
<td>0.00125</td>
</tr>
</tbody>
</table>

Table 33: Time to write and read "swap" file and time read input from incoming request stream

As the tables show, the time to read an input text stream and the time to read a value and write a new value to a swap file is small, even smaller than the measured network times, by another order of magnitude on average. Thus it is possible to marshal ten or more machines quite easily if necessary to increase the load handling capability, without incurring a significant penalty. The reason for a null value is that the action was so quick in some cases that the process (called a thread) measuring time the other (current) process accomplishing the task did not have time to increment, given a time interval of 1/256 of a second. Thus, the penalty incurred in marshalling an incoming stream of requests to various dependent servers is an order of magnitude smaller than the network time necessary to deliver the request.

7.4 Further Research: Improving the system

The research described in this thesis has contributed to the innovation and pioneering work in this field. However, significant areas of research remain. At least four main research areas that are directly related exist. Other derivative research projects such as determination of the cheapest fare with respect to best time might also be feasible should the widespread adoption of the protocol described in this research take hold nationally.
7.4.1 Graphical Interface

A textual interface to the system was concentrated on initially. With the advent of inexpensive Internet access from the home and the need for sophisticated software systems for call centres, an obvious extension of this research is to look at introducing a graphical interface. The main impetus for this is that in a commercial environment, it is necessary to reduce the quantity of time interpreting the desire of the caller and thus reducing the call length. A graphical interface can assist in this task. With the opening up of competition in the supply of graphical images, it is now likely that in the future these call centre features may be available in the home also.

This graphical interface may also assist in overcoming remaining issues with gazetteering, as the presented methodology may not be optimal for all environments. Thus, future research might look at improving the presentation to the user so as to improve usability, i.e. reduce comprehension time.

7.4.2 Response Time

It could be interesting to look at other trials of the SEARCH command to determine if other scenarios show different performance behaviour.

For instance, the relative response time that a sequence command takes with respect to the interval command is interesting to determine if the engine uses one command to calculate the other. Whilst it may be true that the prototype journey search engine in this study did behave in this way, it would be interesting to find out if this is true. However, in a commercial system, it is not envisaged that the journey search engine would use one search process for both situations as it could be argued that this might be inefficient. If so, there may arise a situation where the interval command is sufficiently fast to suggest that the software should use this command for journeys with fewer stages in addition to the more complicated, three or more stage journey.

In another situation, it would be interesting to test the engine for any potential differences between searching for a journey on a bank holiday/weekend or a weekday. If there is a difference in the bus or train service schedules for Sundays and Bank Holidays, then should the engine respond quicker to these types of enquiries, an obvious benefit would result. However, if the services are the same for all days of the week as in some countries (Netherlands), then the engine should respond with the same response time regardless of the day of the week. If the database is large, and the bank holiday journeys are towards the end of the index, then the response time might be longer. The software could be told of this and this information passed on to the traveller so that a longer response time is expected.

A final example of interest could be to determine if the journey time is longer if the journey contains more trips (i.e. more connections). Should a significant difference exist, then it could be beneficial to
break up the request into individual components and request them separately. It is necessary to note that the combined time plus the overhead of splicing it back together must be shorter in time than making the request in its original form to obtain any benefit. However, breaking the journey up might provide for a "hidden journey opportunity" which might not have been otherwise examined. This situation is less likely to arise should walk journeys become more prevalent and detailed.

For speculative research purposes, a selection of journeys was made in serial using software written by the author\(^\text{13}\) to provide insight into what parts of the system were causing the greatest lags in response time. In these analyses, all calls were made remotely using the same 10 MB network, so as to eliminate differences of network response times. Further to other changes, a national coach mode was added using a third implementation of CGI.

<table>
<thead>
<tr>
<th>Journey Planning Task</th>
<th>Test Number</th>
<th>1 sec</th>
<th>2 sec</th>
<th>3 sec</th>
<th>4 sec</th>
<th>5 sec</th>
<th>6 sec</th>
<th>7 sec</th>
<th>8 sec</th>
<th>9 sec</th>
<th>10 sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get Exchange Points</td>
<td></td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Make requests for leg ONE journeys</td>
<td></td>
<td>9</td>
<td>8</td>
<td>4</td>
<td>27</td>
<td>23</td>
<td>7</td>
<td>4</td>
<td>19</td>
<td>22</td>
<td>23</td>
</tr>
<tr>
<td>Make requests for leg TWO journeys</td>
<td></td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>Make requests for leg THREE journeys</td>
<td></td>
<td>22</td>
<td>22</td>
<td>17</td>
<td>13</td>
<td>16</td>
<td>6</td>
<td>19</td>
<td>10</td>
<td>17</td>
<td>17</td>
</tr>
<tr>
<td>Splice journeys together</td>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Total Response Time</td>
<td></td>
<td>36</td>
<td>35</td>
<td>25</td>
<td>45</td>
<td>44</td>
<td>20</td>
<td>31</td>
<td>34</td>
<td>49</td>
<td>48</td>
</tr>
<tr>
<td>Journey Type - Rail matrix</td>
<td></td>
<td>1x2</td>
<td>1x2</td>
<td>1x2</td>
<td>1x2</td>
<td>2x2</td>
<td>1x1</td>
<td>1x1</td>
<td>1x1</td>
<td>1x1</td>
<td>3x2</td>
</tr>
<tr>
<td>- Coach matrix</td>
<td></td>
<td>1x1</td>
<td>1x1</td>
<td>1x1</td>
<td>1x1</td>
<td>1x1</td>
<td>1x1</td>
<td>1x1</td>
<td>1x1</td>
<td>1x1</td>
<td>3x2</td>
</tr>
</tbody>
</table>

Table 34: Response times for individual tasks within journey planning

From the results in table above, it could be seen that one supplier’s journey planners were slower than the others were, and this was affecting the overall response time\(^\text{14}\). Computational time was negligible except where the number of exchange points increased. Even in these cases, the computational time for splicing journeys was difficult to measure due to its brevity. Therefore the principle methods to improve the system lay in reducing the response time of the slowest engine, and in limiting the number of exchange points. However, it is salient that the number of exchange points examined for a particular journey must be sufficient to permit all obvious routes between an origin and a destination. Only local knowledge can do this, assisted by software to optimise the balance between freedom of alternatives and response time to queries.

In summary, an aim of future research could be to consider some of the other methods to minimise amount of time spent waiting for the response from the system by the user as mentioned earlier.

\(^{13}\) Dr. Anthony Lock wrote the splicing subroutine for the local to national to local case.

\(^{14}\) The table has been constructed so as to intentionally mask the suppliers with slower engines.
7.4.3 Error handling

A variety of types of responses need to be handled within the protocol. Not only is there the problem of getting a full, properly formatted response, but there exists also several other cases.

- No response (server down)
- No response (line connection down)
- Busy response (too many connections)
- Incorrect request (improperly formatted request)
- Incorrect login or password (incorrect login or password)
- Incorrect authorisation (computer requesting from this domain is not permitted)

Further, there exist the difficulties in publishing the protocol, and getting the protocol widely accepted by members of the industry. With a wide enough market share or government regulation, it would be possible to encourage or impose the standard upon the industry. The first option is preferred.

7.4.3.1 Specific issues related to the Enquiry Engine and Gazetteer

- no remote database for specified village
- village not listed in specified database
- no places listed for specified village

7.4.3.2 Specific issues related to the Routeing Engine and Registry

- no remote database for specified point
- point not listed in specified database
- no interchanges listed for specified point

It is very important to note that a commercial system should have to have the capability to contend with each of these scenarios. It is especially important to also look at the case where the system will respond to the request, but due to network hold-ups, is not expected to for as long as three seconds. It would then be necessary to look at a method of neatly tidying up previous requests and making a new request at a given interval, say 50% longer than the mean response time, so as not to keep the traveller waiting for what could be time critical information. However, it is not expected that the application of this research will be time-critical.

7.4.4 Data Exchange

An issue arises with respect to the validity of the data being passed. How old or how accurate is the data? Some method of describing the integrity of the data is necessary. Depending upon the actual use of the data, a different format of describing the integrity of the data is necessary. Whilst this study may not appear to concentrate on this issue, it is nevertheless a key issue, which is addressed in further research already underway.
7.4.5 Extension of the Protocol – Fares, ticketing, language, et cetera

It is an obvious extension of this research to look into providing a standard set of commands and tags to enable data regarding fares to be mutually interrogated also. Reservation systems such as Hermes already exist, so in theory the translation of those data standards into the methodologies used herein would be fairly straightforward. However, Hermes is primarily a protocol for rail and ferry services so some consideration is necessary to allow the differences required for bus travel. Tickets cannot always be purchased en-route thus missing connection and similar scenarios need to be addressed. Work by EU-Spirit (2000) may yet some of these differences.

Another consideration, albeit minor, is the consideration of translating foreign characters and text to the requesting system. Whilst not a terribly difficult challenge, it might be a more discussed point of contention than perhaps it may seem to warrant.

7.5 Summary

Of the possible examinations discussed, the research for this thesis concentrated on response time for three of the necessary dialogues (getting places, exchange points and journey segments), and the potential impact of extensibility (exclusive of network delays over long distances or multiple routers).

Analysis of the current enquiry system suggested a range of individual response times. To get places and exchanges, times varied between 0.05 and 0.10 seconds. Trials of the exchange command suggested significant consistency of tests in a local laboratory. To get journey responses, times varied between 0.1 and 0.8 seconds for one or two journeys or up to three seconds for all journeys within the next 24 hours.

With respect to extensibility, analysis of a series of tests suggests that network time is approximately an order of magnitude smaller than the engine response time, and that the time to swap processes is an order of magnitude smaller still. However, it is necessary to remark that occasionally (about once in a thousand) the network time is high (up to three seconds).

Speculative prototyping done in using trial system (explained in Chapter Six) using three counties, two suppliers, and two “national” journey planners (rail and coach) connected via exchange points permitted initial trials of a local-national-local, “cross-boundary” journey enquiry. Times for these tests suggested overall response times between 20 and 50 seconds for three stage (local-national-local) journeys in serial. If the requests are done in parallel, the times can be reduced significantly in complicated cases, depending on journey leg dependencies and sufficient resources within the system (i.e. multiple engines to field parallel search enquiries for each county or region).

For distant-local-only, local-only, or mode-only journeys, response times under a second could be expected or under three seconds for all journeys within the next 24 hours.
From these conclusions, network overheads did not significantly add to the overall journey time for single mode and local journey enquiries. It might be suggested that reduction in the overall response time for complex journey enquiries with multiple modes, and over long distances, is a function of sufficient search engine resources, running requests in parallel, and optimising the number of requests for exchange points.
Chapter 8 - Conclusions

8.1 Background

Although published timetables are available from some local authorities or transport operators, inaccuracies and out-of-date schedules have meant that "word of mouth" remains one of the primary means of obtaining information on public transport services. More recently, desktop publishing has reduced the time necessary to update timetable pamphlets, and decreased the minimum batch size that could be produced economically. A second innovation resulted from changing the manner of data storage used in publishing these pamphlets from simple text files to a tabular or database method of storage. This innovation allowed the possibility to search for details of a particular scheduled service, whilst maintaining the ability to publish schedule booklets, reducing production costs and increasing utilisation.

Public transport operators have recognised the importance of the Internet and have begun to adopt this technology to disseminate schedule information about their services. However, an integrated approach was desired resolve the situation in United Kingdom where many competing and complimentary services exist. This approach needed to cater for different methods of applying Internet technology with different commercial rules and for different modes of public transport, and with a variety of methods to present information. Further, a national system needs to allow for an incremental approach to deployment, as local authorities and service operates are at different stages of implementing "best practice" and thus have different levels of schedule detail in their databases, where such databases existed.

It is apparent that travellers desire an integrated travel itinerary; i.e. a 'one-stop-shop' providing a selection of alternative journeys going from one location to another, instead of having to request one or more pages of schedule data which then have to be individually interpreted. This travel itinerary may cross the boundary of an administrative area, and require a 'home' journey planner to access scheduled data from a 'foreign' journey planner, which may have the data in a different format, and use different technology. Understanding of the fundamental characteristics of an integrated system that interprets proprietary communication methods within a public standard has been a focus of this research.

8.2 Data Requirements

The publication of the TRANSMODEL data dictionary was one of the few significant sources of literature available at the beginning of this research regarding message content. Subsequent review of DATEX, TCIP, RJIS, and meetings with data suppliers Pindar Routel and WS Atkins also assisted in the formulation of message content. Research of available journey planners further contributed to the understanding of the minimum data requirements for people either dependent on public transport for mobility, or who may elect to use public transport in the future. In designing the system, it was decided
to adopt the assumption that the traveller has no knowledge of the local or remote transport services. Bespoke systems could then shortcut superfluous information for individual users with different levels of base knowledge.

Any system needs the traveller to define a set of standard questions. The minimum data requirements of the questions are an origin, a destination, a date and time, and whether this date/time was an 'arrival by time' or a 'departing after time'. For a response or standard answer, the origin, departure time, destination, arrival time, service number, service operator, and mode were the minimum requirements.

Further interviews and research showed that whilst much travel service information was available, this information was in different forms and at different levels of detail. The greatest difficulty in setting up an integrated system was not in finding the information, but of acquiring permission to use the data, relating the data to other sources, and its interpretation by the various parties.

8.3 System Architecture

Of the several approaches to overall system design, two philosophies evolved. The first was to collect, store, and verify all schedule data into a central "warehouse", from which travellers could request all the information needed. However, the cost of this concept to the traveller was high, problems were envisaged with data integrity, and it was doubted whether such a design could withstand the likely load to be placed upon it. This latter point was subsequently demonstrated by the performance of Railtrack's web-based journey planner site.

An alternative philosophy was to store the schedule data locally, as other research had shown that about 80% of requests were likely to be for local schedule data. This philosophy allows the data to be used for a wide variety of purposes, and thus reduces the overall cost of provision whilst benefiting from the advantages of local maintenance. In addition, should details of a particular service be required from another system, these can be requested when necessary and should contain the most current data. Thus, a contribution of this research was to identify whether the second of the above approaches could be made to work efficiently, with advantages of local databases maintained accurately using local knowledge. Furthermore, the distributed database approach removes the necessity to duplicate stored data. This ensures a higher level of data integrity and ease of subsequent expansion of the service.

8.4 Communications and Messaging

The various types and methods of Internet communication were examined under the hypotheses that different journey planner suppliers could communicate with one another via a "trusted third party", an innovative application of the specialist concept of arbitrage, practised in brokerage within the banking sector. The research has shown that, of the numerous network protocols, different software suppliers increasingly accepted TCP/IP as the de facto standard.
Within TCP/IP, several derivative standards exist. This research examined simple mail transfer protocol, file transfer protocol, gopher, veronica, and hyper text transfer protocol as being the most popular protocols most likely to gain wide acceptance. Gopher and veronica were deemed impractical as the technology soon became obsolete or unsupported. Of the three remaining protocols, both FTP and SMTP could have been used, but FTP did not support embedded scripting, and both limited interactions with the user to simple transfer of messages or files. Hyper Text Transfer Protocol (HTTP) allows information to be accessed from databases in addition to both of these functions.

A key feature of HTTP was the ability to stack the system. For example, in HTTP, a user can submit a request to one system, have that receiving system adopt or change the received message as necessary, and then send the changed message to a second system. The second system then receives the message, formulates a reply, and sends it back to the first system. The first system then receives the reply, perhaps changes it in some manner, and then sends it back to the original user. It was concluded, after proving this and the other processes in practice via a series of simple prototypes, that HTTP would be the preferred protocol for allowing communication between two suppliers' systems. This fulfils the requirement of allowing mutual interrogation between journey planners of different suppliers.

Whilst HTTP proved suitable for sending messages between travellers and journey planners, and between journey planners, it became necessary to standardise on a method of sending messages, and a common method of delimiting information within these messages. By extrapolation of HTTP, a specific method delimiting tags for this purpose was created. The essence of this process has since become known as eXtended Markup Language (XML).

8.5 Linking Data Sources

Whilst public transport schedule information has become increasingly available on the Internet, it was tedious and tiresome for the traveller to plan a journey using several different databases. It was concluded that if some method could exist where a journey planner could act on behalf of the traveller and link the individual trips together, a seamless itinerary could be provided. It was concluded that an index could be created to link journey planners, and which could satisfy travellers by creating a reference to a geographic location. By studying the manner in which the Internet works, it was thus concluded that this reference could take on the form of an Internet address, which would then serve as a hook for further information. **An important realisation was the need for a relationship between the administrative area and the schedule database area.** Once this relationship existed, a standard method of transmission and a standard format could be combined to develop a system building on available systems which contained a significant quantity of standardised schedule data in different formats and searched with different routing rules.
8.6 Prototyping the Concepts

Having proved the initial concepts in theory via a series of tests, further development of these theories sought to demonstrate their validity more rigorously. This was done with industrial and government sponsors. A test environment was created in which a traveller could travel wholly within one county or the other, or, using information from a national supplier, he or she could travel between the two counties or to any rail or coach station.

From an internal perspective, the objective was to have two private systems exchange a collection of schedule data using a published communication method. To do this, a series of relationships were built based on the concepts above. First, places were defined as stops. These places were associated with villages or communities in a one-to-many relationship. In the test case, both counties tested had initially contracted out the areas under investigation wholly to one supplier. Thus, a relationship of county to supplier existed. As communities typically fall within one county, the places could then be related to the supplier, providing the hook to further information of exchange points and journeys, which could only be found by requesting information from the suppliers' journey planners. (Further research will be necessary to prove whether these relationships are best suited to all national circumstances).

8.7 Evaluation of the Prototype

In evaluating a prototype system, scheduling data was limited. By measuring the relative magnitude of various components' response times, it was possible to test the system so as to assess the relative impact of earlier decisions. A series of tests were set up to test homogeneous requests for 1) places, 2) exchange points, 3) journeys within one database.

A first analysis determined the approximate number of times the test had to be repeated with the same value to show a stable relationship. It was found that a similar pattern existed both at fifty iterations and at five hundred iterations. Three identical tests, done on two machines with different processor speeds, measured the relative performance of varying the number of places returned within a village. The null response was fastest, with a small number (6) of places at a slightly longer time (0.05 seconds). A larger number of places (=100) took nearly twice as long. As a note, the machine with the slower processor (PII-450) was approximately 10% slower than the Pentium III-500 machine.

Measurements returning exchange points are expected to be about 0.05 seconds, which was confirmed in the results shown. The tests were repeated and the results indicated a similar set of statistics to confirm repeatability. Of particular interest was the relative magnitude of the internet component of the response time. This additional overhead was 0.015 seconds, or about 20% of the overall time. This measurement reflects the relative impact of keeping the exchange points with the schedule data, and not held locally on the machine with the journey navigation software. Thus the overall journey response time is expected to
increase by 0.03 seconds for a three-stage journey. No impact would be had on journeys without exchange points.

Typical response times for the search command ranged from 0.08 seconds to 0.9 seconds for a request to return the next two or three journeys, whereas requests for a whole day's journeys took between one and three seconds. This indicated that for journeys with three or more stages, one might as well request the whole day's journeys if the navigator is capable of making parallel enquiries, both within a stage and overall. Little benefit exists for single database journey requests however, unless the relative time to request a whole day's journeys is decreased.

Several points require consideration. First, the network time, as a proportion of the search engine response time was an order of magnitude smaller than the time taken to search for journeys from the database when requesting the next few journeys. Whilst the network time doubled when requesting a whole day's journeys, the relative magnitude decreased significantly. However, on a frequency of one in every thousand requests, the magnitude of the network time was considerable, measuring nearly three seconds. Whilst uncharacteristic, this second consideration suggests it is salient to have some feature which looks for this regular anomaly. The last consideration is that if computing resources are limited, it may bias the response time for journeys. This consideration has the greatest impact if the national mode journey engines have limited capacity, and the resultant impact will be felt primarily on those journeys that depend heavily on national mode journeys. Therefore, national services such as rail, coach and air may need more computing resources than currently exist to cope with expected request volumes even though their databases are 'small' and thus have a short response time.

Finally, the relative impact of marshalling a request within a local area network is small. Whilst it was difficult to get a precise measurement, it is approximately an order of magnitude smaller than the network overhead time. Thus the additional penalty of about 0.003 seconds with respect to response time for adding extra computing resources is much smaller than the relative benefit of additional capacity of handling more requests at once. With respect to extensibility, the relative penalty of splitting the system is similar to the time to look up a few places or exchange points. This measurement was about 0.05 seconds. As a result, the penalty for distributing the system is small in comparison to creating a single large database and searching through that, whilst ensuring the ability to interconnect with other national systems.

8.8 Further Research

Whilst it is noted that this research has made a substantial contribution to the introduction of a practical, integrated journey planning system for the UK, further research is recommended in the following areas to resolve remaining issues of fine-tuning a journey planning system in practice.
1. Optimisation of the response time of a journey planner navigator with respect to journey search engine size and number of exchange points, taking into account the effects of loading (multiple, concurrent journey requests), and including delays induced from network traffic/packet propagation, for a call centre and web based clientele.

2. Confirmation of the indicators provided herein using schedule data from other suppliers' engines, and tested to measure the impact of databases with significantly different sizes (i.e. a large database for a dense urban area with many services in contrast with a small rural county database).

3. Examination of methods to methodologically determine the most inexpensive or shortest time journeys, once the introduction of fares, reservations and ticketing have been added to the protocol.

4. Examination of the practicality of the itinerary from the journey planner versus what the traveller would expect, so as to add comprehensive walk, cycle or taxi links.

5. Examination of the relative effect of improvements to the graphical interfaces on perception and understanding.

In summary, a collection of Internet-based methods that are analogous to what is now XML has been independently developed and prototyped. They allow electronic journey planners to remotely interrogate public transport schedule information held within compliant journey planners using a common protocol. This feature is one method which could enable governments to link existing systems whilst allowing for new and existing suppliers of information to extend the network of coverage by adding journey planners from new areas or regions. Whilst the conclusions in this research indicate the relative importance of a limited number of specific characteristics for a national system, several issues remain regarding the wide distribution of integrated, high quality, timely, multiple mode, journey planning information for public transport using an Internet-based system.
References


- DETR (a) *Better Information for Bus Passengers - A guide to Good Practice,* Buses and Taxis Division, Zone 3/13, Great Minster House, 76 Marsham Street, London 1996, DETR SWIP 4DR. Tel. orderline: 0171 676 2292

- DETR (b). "Climate Change Report Presented to the Prime Minister". Nation Press Office. 31 March 1998


- DETR (h) Transport Bill. The Stationery Office National Publishing. Last Updated: 12 Jan 2001


157


• Funnell, R. Telephone and Personal Interviews. From 1996 to Present.


• Guardian. http://www.educationunlimited.co.uk/. Link to article can no longer be found.


• Houghton, P. Personal and Telephone Interviews. Pindar at York, Derby, and Princes Risborough. Also at Southampton University. 1995 through to 1999.


158


• Marshall, B. Webmaster of StationMaster website. Personal communications from 1997.

• Mason, J. Personal Interview. Fleet (August) and Bournemouth (August). 1996.


• McCormack, J.E. The specification of trip planning systems: Report on the development of the trip planning system prototype, University of Leeds, School of Computer Studies, 94.5, June, 1994. SERC GR/H/15745.


• Mouskos, Kyriacos et al "Toward a Multimodal Advanced Traveller Information System (MATIS)", VERTIS Ed. Proceedings of the Second World Congress on Intelligent Transport Systems, '95 Yokohama, pg. 715

• Multimap. "A complete interactive atlas on the Web". http://uk.multimap.com/map/places.cgi


• OV Reisinformatie V.O.F. "OVR" (Public Transport Travel Information). NL 3511 DT Utrecht. The Netherlands. Telephone: 31 30 328806

• Potgraven, Patrick (Directorate-General for Public Works, highway Division). Personal correspondence and enclosures. 26 January 1996.


• Roach, Helen. Telephone Interview. Formerly at Leeds University on the EUROBUS project, 1996.

• ROMANSE. [http://www.romase.org.uk](http://www.romase.org.uk) (original documentation no longer available on-site)


• Smith, B. (Associate Professor at Virginia Tech.) Non-published presentation at Transportation Research Board. January 12, 2000.


• Personal Interview. Transport Planner (Public Transport) of Southampton City Council at Southampton City Council offices, November 1994.


• ST-ITS-TCIP-SCH. Transit Communications Interface Profile, Standard on Scheduling/Runcutting (SCH) Objects, Version 1.0, Institute of Transportation Engineers, February 20, 1998

• TV Licencing, 1997-1999.

• Tognoni, G. The intermodal information: Objectives and progress of the European project INFOTEN. Proceedings of the Third World Congress of Intelligent Transport Systems, ed. VERTIS, 1996.


• Tyler, Nick (a). Accessible Public Transport in Rural Areas. Proceedings from the Rural Transport for All seminar, Lancaster University, July., 1999


Bibliography

- “Consultants appointed for London airport access study”, Local Transport Today, Issue 167, 31 August 1995. pg. 2
- “It is important to have both the time and the price right next to each other.” Statement of James Collins, Director of ThamesLink, British Rail, upon reviewing an interactive, trial demonstration of this research, May 11, 1995 at Southampton University.
- “The Future of Road Transport”, inaugural address, Prof. Mike McDonald, Professor of Civil Engineering, Southampton University, May 17, 1995.
- Charlier, Jacques, “Sur l’importance des temps et des couts d’access aux gares et aux aeroports dans le jeu de la concurrence air-fer: L’exemple des relations intermetropolitaines en Allemagne

163

- Chartered Institute of Transport Monthly meeting, Southampton Institute, October 26, 1995.
- Department of Commerce and Trade: U.S. Supreme Court case of People verses U.S. Airlines, 1994
- Dr Schoon’s reference - Walking as a mode of transport, TRB, Washington, D.C. 1995
- Editor, “The Government Policy that was lost in Transit”, Independent, 2 Dec. 1995
- EUROBUS, Deliverable #6, Cete Mediterranien, 1993.
- Southampton Airport International, Ltd, Groundside Operations, Telephone 44 1703 620021

164
- Hughes, Peter, *Transport telematics make their mark - but who really gets the benefits?*, Local Transport Today, 14 September, 1995, pg. 12, 13.


• Rail Privatisation, deregulation, and open access, The Institute of Civil Engineers, Thomas Telford Services, Infrastructure Policy Group, Tuesday, March 10, 1992. ISBN: 0 7277 19149


• Sanger, “City Centre Transport plan. Keeping Your City on the Move.”, Southampton City Council presentation of the fifteen year land-use plan, 14 Nov. 1995 at Southampton University.


• Steers, Jim, at the Chartered Institute of Transport’s “Great Transport Debate” in Basingstoke, UK, Feb. 8, 1996.


166
• Draft IEEE P1489/D0.0.7, Draft Standard for Data Dictionaries for Intelligent Transportation Systems, Version 0.0.7, October 9, 1997
• Draft IEEE P1488/D0.0.6, Draft Standard for Message Set Template for Intelligent Transportation Systems, Version 0.0.6, October 17, 1997
• ISO/IEC 8824:1994, Abstract Syntax Notation One (ASN.1)
• USPS Postal Addressing Standards, Publication 28, November 1997
• ISO 31/0. 1981. General principles concerning quantities, units and symbols.
• ISO 6937/2. 1983. Information processing - Coded character sets for text communication.
• ISO 7372. 1986. Trade Data Elements Directory.
• ISO 8859. 1987. Information processing - 8-bit single byte coded graphic character sets.
Appendices

A - Figures illustrating public transport related information on the Internet..............169
B - Table of Internet addresses of surveyed journey planners...................................173
C - An Extract of Datex Data Definitions for Public Transport ..................................175
D - Software code for a sample ECHO server in Visual Basic © Microsoft................182
E - JourneyWeb protocol as tested in this research (Ver. 1.0)....................................187
F - Library of common subroutines for using XML.........................................................211
G - Testing software used in this thesis.......................................................................217
Appendix A

The following sections depict a series of images collected from the Internet which provide samples of public transport resources on the Internet matching the descriptions provided in the text.

Routes, Maps, and Weather Information

Figure 1 : Route Map of Southampton University's UNI Link Network

Figure 2 : Map of road network with the various campus sites

Figure 3 : Sample of a weather map used for advising travellers

Facilities Information
The research that produced the figure below aimed at providing information including other facilities in greater detail, such as restrooms, lifts, grills, news agents, cash point machines, and so forth. However, whilst these are important, samples of these are not yet available.

**Newcastle Central Station# (m)**

![Facilities Available at Newcastle Central Station](image)

**Figure 4**: Facilities available at Newcastle Central (rail) Station

**Schedule Information**

The following sample shows a digital copy of the timetable for service 3/9 between Stoke Mandeville, Aylesbury, and Haydon Hill, all in Buckinghamshire, England. The table was produced from a database by a scripting software designed specifically for this purpose. As changes are made to the database by the county council, the changes are automatically reflected in the displayed page available to potential travellers. This feature is not available in the printed timetables sold or distributed to users and homeowners.
3, 9 : Stoke Mandeville Hospital - Town Centre - Haydon Hill

Timetable from 5 September 1999
Mondays to Fridays
Service Number
Operator
Days of operation
Stoke Mandeville Hosp
Halton Court Hand Sch
Hawksiade Farm Estate
Walton Court Shops
Southcourt Church Sq
Southcourt Chestnut Cr
Oxford Rd (Ayles)
Aylesbury Town Centre
Dunsham Lane
Headowcroft (Ayles)
Quarrendon St Peter’s
Quarrendon Holland Rd
Edison Road
Haydon Hill
Quarrendon Jackson Rd
Quarrendon J Kennedy

9 9 9
0614 0644 0615 0615 0645 0645 0618 0618 0648 0648 0621 0621 0651 0651
0655 0658 0701 0701 0724 0724 0705 0705 0746 0746 0737 0737
0710 0721 0725 0725 0731 0731 0751 0751 0737 0737
0730 0742 0738 0738 0744 0744
0731 0741 0745 0745
0735 0742 0746 0746
0741 0746 0746

Fare Information
Fare information is less frequently found on the Internet, especially in England due to the intense competition amongst bus companies. However, pressures are encouraging greater availability of this information. In the United States, the table reproduced in this figure provides a static listing of fairs by zone for travellers.

Caltrain stations are divided into 9 fare zones.

Fares by Number of Zones in Journey
(i.e. a trip from Zone SF to Zone 3 would be 4 zones - SF, 1, 2, 3)

<table>
<thead>
<tr>
<th>Zones</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>GW</td>
<td>$1.25</td>
<td>$2.00</td>
<td>$2.75</td>
<td>$3.25</td>
<td>$4.00</td>
<td>$4.75</td>
<td>$5.25</td>
<td>$6.00</td>
<td>$6.75</td>
</tr>
<tr>
<td>S/D/C</td>
<td>0.50</td>
<td>1.00</td>
<td>1.25</td>
<td>1.50</td>
<td>2.00</td>
<td>2.25</td>
<td>2.50</td>
<td>3.00</td>
<td>3.25</td>
</tr>
<tr>
<td>OW</td>
<td>1.00</td>
<td>1.60</td>
<td>2.00</td>
<td>2.50</td>
<td>3.00</td>
<td>3.50</td>
<td>4.00</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>1D</td>
<td>11.25</td>
<td>17.00</td>
<td>22.75</td>
<td>28.50</td>
<td>34.00</td>
<td>39.75</td>
<td>45.50</td>
<td>51.25</td>
<td>56.75</td>
</tr>
<tr>
<td>RD</td>
<td>35.50</td>
<td>53.25</td>
<td>70.75</td>
<td>88.50</td>
<td>106.25</td>
<td>124.00</td>
<td>141.75</td>
<td>159.50</td>
<td>177.25</td>
</tr>
<tr>
<td>TBH</td>
<td>34.75</td>
<td>52.25</td>
<td>69.75</td>
<td>87.00</td>
<td>104.25</td>
<td>121.75</td>
<td>139.00</td>
<td>156.50</td>
<td>173.75</td>
</tr>
<tr>
<td>DR</td>
<td>26.75</td>
<td>40.00</td>
<td>53.50</td>
<td>66.75</td>
<td>80.25</td>
<td>93.80</td>
<td>107.00</td>
<td>120.25</td>
<td>133.75</td>
</tr>
</tbody>
</table>

Ticket Types

<table>
<thead>
<tr>
<th>GW</th>
<th>One Way</th>
</tr>
</thead>
<tbody>
<tr>
<td>S/D/C</td>
<td>Senior/Disable/Child</td>
</tr>
<tr>
<td>OW</td>
<td>One-way Discount</td>
</tr>
<tr>
<td>1D</td>
<td>1D-ride Ticket</td>
</tr>
<tr>
<td>NO</td>
<td>Monthly Ticket</td>
</tr>
<tr>
<td>7EM</td>
<td>Ticket by Rail</td>
</tr>
<tr>
<td>DR</td>
<td>Discount Monthly</td>
</tr>
</tbody>
</table>

- Round Trip fare is two times the one-way fare.
- $1 surcharge on all tickets purchased on train when station is open or ticket vending machines are available.
- A transfer from Gilroy trains will be permitted at the San Jose Diridon Station only, if the originating train does not stop at the customer’s desired destination due to scheduling. Request a transfer from the conductor.

Figure 6 : Fair Zones in San Francisco
Journey Planning Information

This figure provides a summary of a journey from Aylesbury to Oxford.

### Journey Departing at 11:30 Arriving at 12:47

<table>
<thead>
<tr>
<th>Depart/Arrive</th>
<th>Time</th>
<th>Place Name</th>
<th>Location</th>
<th>Operator</th>
<th>Service</th>
<th>Est. Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Depart</td>
<td>11:30</td>
<td>Aylesbury Town Centre</td>
<td>Bus Station Bay 9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Arrive</td>
<td>12:47</td>
<td>Oxford</td>
<td>St Aldates</td>
<td></td>
<td>280</td>
<td>1 Hour 17 Mins</td>
</tr>
</tbody>
</table>

Total Journey Time: 1 Hour 17 Minutes

*Back To Summary Table*

Figure 7: Sample Journey Planner output
Appendix B

This table lists the findings of a survey to find trip or journey planners on the Internet.

<table>
<thead>
<tr>
<th>URL</th>
<th>Name</th>
<th>Origin</th>
<th>Origin Type</th>
<th>Destination</th>
<th>Destination Type</th>
<th>Date</th>
<th>Date Type</th>
<th>Time</th>
<th>TimeType</th>
<th>Dep/Arr</th>
<th>Via</th>
<th>Language</th>
<th>Exc</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://romanse.soton.ac.uk/">http://romanse.soton.ac.uk/</a></td>
<td>WWW-ROMANSE</td>
<td>y</td>
<td>Key-match</td>
<td>y</td>
<td>key-match</td>
<td>y</td>
<td>Key-match</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>Y</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Kiosk-ROMANSE</td>
<td>y</td>
<td>Key-match</td>
<td>y</td>
<td>key-match</td>
<td>y</td>
<td>Key-match</td>
<td>y</td>
<td>n</td>
<td>n</td>
<td>Y</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td><a href="http://www.railtrack.co.uk/travel/">http://www.railtrack.co.uk/travel/</a></td>
<td>WWW-RailTrack</td>
<td>y</td>
<td>Key-match</td>
<td>y</td>
<td>key-match</td>
<td>dd/mm/yyyy</td>
<td>y</td>
<td>y</td>
<td>n</td>
<td>y</td>
<td>N</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td><a href="http://www.nationalexpress.co.uk/natxbok1.htm">http://www.nationalexpress.co.uk/natxbok1.htm</a></td>
<td>WWW-National Express</td>
<td>y</td>
<td>Key-match</td>
<td>y</td>
<td>key-match</td>
<td>y</td>
<td>Select</td>
<td>y</td>
<td>n</td>
<td>n</td>
<td>N</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td></td>
<td>WWW-Cumbria</td>
<td>y</td>
<td>Select</td>
<td>y</td>
<td>select</td>
<td>y</td>
<td>Select</td>
<td>y</td>
<td>n</td>
<td>n</td>
<td>N</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td></td>
<td>WWW-Cumbria</td>
<td>y</td>
<td>Key-match</td>
<td>y</td>
<td>key-match</td>
<td>y</td>
<td>ddmmyyyy</td>
<td>y</td>
<td>Hhmm</td>
<td>dep</td>
<td>Y</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td></td>
<td>WWW-Hertfordshire</td>
<td>y</td>
<td>Select</td>
<td>y</td>
<td>select</td>
<td>y</td>
<td>Select</td>
<td>y</td>
<td>n</td>
<td>n</td>
<td>N</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td></td>
<td>WWW-Hertfordshire</td>
<td>y</td>
<td>Key-match</td>
<td>y</td>
<td>key-match</td>
<td>y</td>
<td>ddmmyyyy</td>
<td>y</td>
<td>Hhmm</td>
<td>dep</td>
<td>Y</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td></td>
<td>WWW-Lancashire</td>
<td>y</td>
<td>Select</td>
<td>y</td>
<td>select</td>
<td>y</td>
<td>Select</td>
<td>y</td>
<td>n</td>
<td>n</td>
<td>N</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td></td>
<td>WWW-Lancashire</td>
<td>y</td>
<td>Key-match</td>
<td>y</td>
<td>key-match</td>
<td>y</td>
<td>ddmmyyyy</td>
<td>y</td>
<td>Hhmm</td>
<td>dep</td>
<td>Y</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td><a href="http://www.sncf.fr/cgi-bin/rho/rho?dh=s11">http://www.sncf.fr/cgi-bin/rho/rho?dh=s11</a></td>
<td>WWW-SNCF</td>
<td>y</td>
<td>Key-match</td>
<td>y</td>
<td>key-match</td>
<td>dd/mm/yyyy</td>
<td>y</td>
<td>HH:mm</td>
<td>dep</td>
<td>n</td>
<td>Y</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td><a href="http://bahn.hafas.de/bin/db.s97/detect.exe/bin/db.s97/query.exe/e">http://bahn.hafas.de/bin/db.s97/detect.exe/bin/db.s97/query.exe/e</a>?</td>
<td>WWW-DB</td>
<td>y</td>
<td>Key-match</td>
<td>y</td>
<td>key-match</td>
<td>dd/mm/yyyy</td>
<td>y</td>
<td>HH:mm</td>
<td>y</td>
<td>y</td>
<td>Y</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td><a href="http://www.renfe.es/ic/index.htm">http://www.renfe.es/ic/index.htm</a></td>
<td>WWW-RENFÉ</td>
<td>y</td>
<td>Select</td>
<td>y</td>
<td>select</td>
<td>n</td>
<td>n</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><a href="http://www.clubieire/RailNet/map/">http://www.clubieire/RailNet/map/</a></td>
<td>WWW-IE</td>
<td>y</td>
<td>Map</td>
<td>y</td>
<td>map</td>
<td>y</td>
<td>dayofweek</td>
<td>select</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td><a href="http://www.clubieire/RailNet/map/">http://www.clubieire/RailNet/map/</a></td>
<td>WWW-IE</td>
<td>y</td>
<td>Select</td>
<td>y</td>
<td>select</td>
<td>y</td>
<td>dayofweek</td>
<td>select</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td><a href="http://www.vr.fi/heo/aikae.htm">http://www.vr.fi/heo/aikae.htm</a></td>
<td>WWW-SWEDEN</td>
<td>y</td>
<td>key-match</td>
<td>y</td>
<td>key-match</td>
<td>y</td>
<td>yymmd</td>
<td>hh:mm</td>
<td>y</td>
<td>y</td>
<td>Y</td>
<td>Y</td>
<td></td>
</tr>
</tbody>
</table>

URL                      | Name                   | Origin | Origin Type | Destination | Destination Type | Date       | Date Type | Time    | TimeType | Dep/Arr | Via | Language | Exc |
|--------------------------|------------------------|--------|-------------|-------------|------------------|------------|-----------|---------|----------|---------|-----|----------|-----|

173
<table>
<thead>
<tr>
<th>URL</th>
<th>Country</th>
<th>Feature</th>
<th>Key</th>
<th>Format</th>
<th>Date</th>
<th>Time</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.ifi.uio.no/~terje/rail/timetab/index.html#TIMETABLES">http://www.ifi.uio.no/~terje/rail/timetab/index.html#TIMETABLES</a></td>
<td>WWW-Norway</td>
<td>Timetables by Service</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>n</td>
</tr>
<tr>
<td><a href="http://www.rejseplan.dk/rejseplan_uk.html">http://www.rejseplan.dk/rejseplan_uk.html</a></td>
<td>WWW-DK</td>
<td></td>
<td>Y</td>
<td>key-match</td>
<td>y</td>
<td>ddmmyy</td>
<td>hhmm</td>
</tr>
<tr>
<td><a href="http://www.ns.nl/reisinfor/index.html">http://www.ns.nl/reisinfor/index.html</a></td>
<td>WWW-Netherlands</td>
<td></td>
<td>Y</td>
<td>key-match</td>
<td>y</td>
<td>ddmmyy</td>
<td>hhmm</td>
</tr>
<tr>
<td><a href="http://www.cp.pt/horarios/e_index.html">http://www.cp.pt/horarios/e_index.html</a></td>
<td>WWW-Portugal</td>
<td>Timetables by Service</td>
<td>Y</td>
<td>key-match</td>
<td>y</td>
<td>ddmmyy</td>
<td>hhmm</td>
</tr>
<tr>
<td><a href="http://www.ose.gr/ose1/home2.html">http://www.ose.gr/ose1/home2.html</a></td>
<td>WWW-Greece</td>
<td>Timetables and Fares by Service</td>
<td>Y</td>
<td>key-match</td>
<td>n</td>
<td>y</td>
<td>yymmdd</td>
</tr>
<tr>
<td><a href="http://idos.cit.cdrail.cz/cgi-bin/dos1.pl/adjprj/el">http://idos.cit.cdrail.cz/cgi-bin/dos1.pl/adjprj/el</a></td>
<td>WWW-Czech</td>
<td></td>
<td>Y</td>
<td>key-match</td>
<td>n</td>
<td></td>
<td>yymmdd</td>
</tr>
<tr>
<td><a href="http://metro.jussieu.fr:10001/bin/select/english/united-kingdom/london">http://metro.jussieu.fr:10001/bin/select/english/united-kingdom/london</a></td>
<td>WWW-MetroNavigator</td>
<td></td>
<td>Y</td>
<td>key-match</td>
<td>y</td>
<td></td>
<td>key-match</td>
</tr>
<tr>
<td><a href="http://www.pindar.co.uk:80/cgi-bin/jp.cgi?command=index">http://www.pindar.co.uk:80/cgi-bin/jp.cgi?command=index</a></td>
<td>WWW-Bucks</td>
<td></td>
<td>Y</td>
<td>select</td>
<td>y</td>
<td></td>
<td>select</td>
</tr>
</tbody>
</table>
Appendix C

Datex Data Definitions

Introduction: Correspondence of the Dictionary Items with Transmodel

The present section describes the correspondence between the items of the dictionary referring to Public Transport and the reference data model for Public Transport operation, Transmodel.

Only the Transmodel concepts of interest in the context of the dictionary have been included. For instance, all the entities describing the basic topology of the network are not included in the dictionary: the entity POINT is only described by two of its subtypes, while the entity LINK is not present at all.

Some concepts which are described in Transmodel by several detailed entities have been aggregated in the dictionary for simplicity reasons. For instance, the numerous entities describing the various possibilities to define scheduled run times have been grouped into one main concept.

The definition of entities has been sometimes adapted as necessary, to be understandable in the context of the dictionary. Attributes have been defined explicitly, which is not the case in the Transmodel documentation.

The last section of this appendix provides the relationship between some items of the dictionary referring to Public Transport and Transmodel concepts, included or not in the dictionary.

POINTS

Stop Point: A point where passengers can board or alight from PT Vehicles.
   corresponds to the Transmodel entity STOP POINT
   (same definition)
Stop Point Name: The name of a Stop Point.
   corresponds to the attribute Name of POINT
Stop Point for Boarding: A Stop Point used for boarding.
   corresponds to the attribute For Boarding of STOP POINT
Stop Point for Alighting: A Stop Point used for alighting.
   corresponds to the attribute For Alighting of STOP POINT
PT Timing Point: A point against which the timing information necessary to build PT schedules may be recorded.
   corresponds to the Transmodel entity TIMING POINT
   (same definition)
PT Timing Point Name: The name of a PT Timing Point.
   corresponds to the attribute Name of POINT
Stop Area: A group of Stop Points close to each other.
   corresponds to the Transmodel entity STOP AREA
   (same definition)
Stop Area Name: The name of a Stop Area.
   corresponds to the attribute Name of STOP AREA

ROUTES

PT Route: An ordered list of points, defining one single path for PT services through the road (or rail) network.
   corresponds to the Transmodel entity ROUTE
   (definition slightly adapted)
PT Route Name: The name or number of a PT Route.
corresponds to the Transmodel entity DIRECTION

PT Terminus: The end of a PT Route.
  corresponds to a general definition of a Transmodel Glossary item

Journey Pattern: An ordered list of Stop Points and PT Timing Points on a single PT Route, describing the pattern of working for PT Vehicles.
  corresponds to the Transmodel entity JOURNEY PATTERN

Journey Pattern Name: The name or number of a Journey Pattern.
  corresponds to the attribute Name of JOURNEY PATTERN

Point in Journey Pattern: A Stop Point or a PT Timing Point with its order in that Journey Pattern.
  corresponds to the Transmodel entity POINT IN JOURNEY PATTERN

Journey Pattern Origin: The first Point in Journey Pattern of that Journey Pattern.
  corresponds to the attribute Order of POINT IN JOURNEY PATTERN

Journey Pattern Destination: The last Point in Journey Pattern of that Journey Pattern.
  corresponds to the attribute Order of POINT IN JOURNEY PATTERN

Destination Display: The advertised destination of a specific Journey Pattern.
  corresponds to the Transmodel entity DESTINATION DISPLAY

PT Common Section: A part of a PT network where the PT Routes of several Journey Patterns are going in parallel and where the synchronisation of PT Vehicle Journeys is planned and controlled.
  corresponds to the Transmodel entity COMMON SECTION

PT Line: A group of PT Routes generally known to the public by a similar name or number.
  corresponds to the Transmodel entity LINE

PT Line Name: The name or number of a PT Line.
  corresponds to the attribute Name of LINE

DAYS

PT Day Type: A type of day characterised by one or more properties which affect PT operation (e.g. weekday in school holidays).
  corresponds to the Transmodel entity DAY TYPE

PT Property of Day: A property which a day may possess, affecting PT operation, such as school holiday, weekday, summer, winter, etc.
  corresponds to the Transmodel entity PROPERTY OF DAY

PT Calendar Day: A specific day in the calendar where PT operation takes place.
  corresponds to the Transmodel entity CALENDAR DAY

VEHICLES

PT Vehicle: An individual PT vehicle used for carrying passengers.
  corresponds to the Transmodel entity VEHICLE

PT Vehicle Type: A type of PT Vehicle.
  corresponds to the Transmodel entity VEHICLE TYPE

PT Vehicle Seating Capacity: The seating capacity of a PT Vehicle Type.
  corresponds to the attribute Seating Capacity of VEHICLE TYPE

PT Vehicle Standing Capacity: The standing capacity of a PT Vehicle Type.
corresponds to the attribute Standing Capacity of VEHICLE TYPE

PT Vehicle Special Place Capacity : The special place capacity of a PT Vehicle Type.
corresponds to the attribute Special Place Capacity of VEHICLE TYPE

JOURNEYS

Journey : The movement of a PT Vehicle between two specified points, e.g. two Termini.
corresponds to a general definition of a Transmodel Glossary item

PT Service : An advertised collection of PT Journeys between two Termini, possibly with variations.
corresponds to a general definition of a Transmodel Glossary item

PT Vehicle Journey : A particular journey of a PT Vehicle on a particular PT Day Type.
corresponds to the Transmodel entity VEHICLE JOURNEY

(same definition)

PT Departure Time : The departure time of a specific PT Vehicle Journey.
corresponds to the attribute Departure Time of VEHICLE JOURNEY

PT Dated Vehicle Journey : A particular journey of a PT Vehicle on a particular PT Calendar Day, including all modifications possibly decided by the control staff.
corresponds to the Transmodel entity DATED VEHICLE JOURNEY

(same definition)

RUN TIMES

PT Run Time : The planned time for a PT Vehicle to traverse a link between two PT Timing Points.
corresponds to an aggregation of the Transmodel entities VEHICLE JOURNEY RUN TIME, JOURNEY PATTERN RUN TIME, DEFAULT SERVICE JOURNEY RUN TIME, DEFAULT DEAD RUN TIME

(definition adapted accordingly)

PT Run Time Duration : The duration of a PT Run Time.
corresponds to the attribute Duration of the aggregated RUN TIME entities

PT Wait Time : The planned time a PT Vehicle has to wait at a specific PT Timing Point.
corresponds to an aggregation of the Transmodel entities VEHICLE JOURNEY WAIT TIME, JOURNEY PATTERN WAIT TIME

(definition adapted accordingly)

PT Wait Time Duration : The duration of a PT Wait Time.
corresponds to the attribute Duration of the aggregated WAIT TIME entities

PT Mean Run Time : An estimated value of the mean run time between two PT Timing Points, used to inform passengers on the mean duration of Trips.
corresponds to the Transmodel entity MEAN RUN TIME

(definition slightly adapted)

PT Mean Run Time Duration : The duration of a PT Mean Run Time.
corresponds to the attribute Duration of MEAN RUN TIME

PT Mean Passenger Wait Time : An estimated mean waiting time for a passenger at a Stop Point, used to calculate the approximate duration of a Trip.
corresponds to the Transmodel entity MEAN PASSENGER WAIT TIME

(same definition)

PT Mean Passenger Wait Time Duration : The duration of a PT Mean Passenger Wait Time.
corresponds to the attribute Duration of MEAN PASSENGER WAIT TIME

PASSING TIMES

PT Passing Time : Time data concerning PT Vehicles passing a particular point (e.g. arrival time, departure time, waiting time).
corresponds to the Transmodel entity PASSING TIME

(same definition)

Timetabled Passing Time : Long-term planned time data concerning PT Vehicles passing a particular Point in Journey Pattern, on a specific PT Vehicle Journey, for a specified PT Day Type.
corresponds to the Transmodel entity TIMETABLED PASSING TIME
**Timetabled Arrival Time**: The arrival time of a Timetabled Passing Time. 
corresponds to the attribute Timetabled Arrival Time of TIMETABLED PASSING TIME

**Timetabled Departure Time**: The departure time of a Timetabled Passing Time. 
corresponds to the attribute Timetabled Departure Time of TIMETABLED PASSING TIME

**Timetabled Waiting Time**: The waiting time of a Timetabled Passing Time. 
corresponds to the attribute Timetabled Waiting Time of TIMETABLED PASSING TIME

**Target Passing Time**: Time data about when a PT Vehicle should pass a particular Point in Journey Pattern, on a particular PT Dated Vehicle Journey, according to the latest valid plan. It is the latest revision of a Timetabled Passing Time. 
corresponds to the Transmodel entity TARGET PASSING TIME

**Target Arrival Time**: The arrival time of a Target Passing Time. 
corresponds to the attribute Aimed Arrival Time of TARGET PASSING TIME

**Target Departure Time**: The departure time of a Target Passing Time. 
corresponds to the attribute Aimed Departure Time of TARGET PASSING TIME

**Target Waiting Time**: The waiting time of a Target Passing Time. 
corresponds to the attribute Aimed Waiting Time of TARGET PASSING TIME

**Estimated Passing Time**: Calculated dated time data about when a PT Vehicle will pass a particular Point in Journey Pattern, on a specific PT Dated Vehicle Journey. It is used to inform passengers, and may be different of the Target Passing Time. 
corresponds to the Transmodel entity ESTIMATED PASSING TIME

**Estimated Arrival Time**: The arrival time of an Estimated Passing Time. 
corresponds to the attribute Expected Arrival Time of ESTIMATED PASSING TIME

**Estimated Departure Time**: The departure time of an Estimated Passing Time. 
corresponds to the attribute Expected Departure Time of ESTIMATED PASSING TIME

**Estimated Waiting Time**: The waiting time of an Estimated Passing Time. 
corresponds to the attribute Expected Waiting Time of ESTIMATED PASSING TIME

**Recorded Passing Time**: The actual passing of a PT Vehicle at a pre-defined point during a PT Dated Vehicle Journey (e.g. the passing detected by a vehicle location tracking system). 
corresponds to the Transmodel entity RECORDED PASSING TIME

**Recorded Arrival Time**: The arrival time of a Recorded Passing Time. 
corresponds to the attribute Actual Arrival Time of RECORDED PASSING TIME

**Recorded Departure Time**: The departure time of a Recorded Passing Time. 
corresponds to the attribute Actual Departure Time of RECORDED PASSING TIME

**Recorded Waiting Time**: The waiting time of a Recorded Passing Time. 
corresponds to the attribute Actual Waiting Time of RECORDED PASSING TIME

**Recorded Stop**: The recorded stop of a PT Vehicle at a Stop Point during actual PT operation, to possibly let passengers board or alight the vehicle. 
corresponds to the Transmodel entity RECORDED STOP

**INTERCHANGES**

**Connection**: The association of two or more PT Lines at a Stop Point or at close Stop Points, allowing possible Interchanges for passengers. 
corresponds to a general definition of a Transmodel Glossary item

**Transfer**: The transfer of passengers from one PT Vehicle to another. The Transfer may occur for an Interchange or for operational reasons, on request of the staff. 
corresponds to a general definition of a Transmodel Glossary item

**Interchange**: The Transfer of passengers from one PT Line to another at a Connection point, when no direct line is available towards their destination. 
corresponds to a general definition of a Transmodel Glossary item

**Connection Link**: The physical (spatial) possibility for a passenger to change from one PT Vehicle to another to continue a Trip. 
corresponds to the Transmodel entity CONNECTION LINK

(definition slightly adapted)
Connection Link Distance: The distance for a Connection Link.  
  corresponds to the attribute Distance of CONNECTION LINK

Connection Link Default Time: The default duration of a Connection Link.  
  corresponds to the attribute Default Time of CONNECTION LINK

Connection Link Frequent Traveller Time: The duration of a Connection Link for a frequent traveller.  
  corresponds to the attribute Frequent Traveller Time of CONNECTION LINK

Connection Link Occasional Traveller Time: The duration of a Connection Link for an occasional traveller.  
  corresponds to the attribute Occasional Traveller Time of CONNECTION LINK

Connection Link Mobility Restricted Time: The duration of a Connection Link for a mobility restricted traveller.  
  corresponds to the attribute Mobility Restricted Traveller Time of CONNECTION LINK

Connection Link Mobility Restricted Possibility: The possibility to traverse a Connection Link for a mobility restricted traveller.  
  corresponds to the attribute Suitable for Mobility Restricted of CONNECTION LINK

Access Link: The physical (spatial) possibility for a passenger to access or leave the PT system. This link may be used during a Trip from a place (origin of the Trip) to a Stop Point, or from a Stop Point to a place (destination of the Trip).  
  corresponds to the Transmodel entity ACCESS LINK  
  (same definition)

Access Link Distance: The distance for an Access Link.  
  corresponds to the attribute Distance of ACCESS LINK

Access Link Default Time: The default duration of an Access Link.  
  corresponds to the attribute Default Time of ACCESS LINK

Access Link Frequent Traveller Time: The duration of an Access Link for a frequent traveller.  
  corresponds to the attribute Frequent Traveller Time of ACCESS LINK

Access Link Occasional Traveller Time: The duration of an Access Link for an occasional traveller.  
  corresponds to the attribute Occasional Traveller Time of ACCESS LINK

Access Link Mobility Restricted Time: The duration of an Access Link for a mobility restricted traveller.  
  corresponds to the attribute Mobility Restricted Traveller Time of ACCESS LINK

Access Link Mobility Restricted Possibility: The possibility to traverse an Access Link for a mobility restricted traveller.  
  corresponds to the attribute Suitable for Mobility Restricted of ACCESS LINK

Journey Pattern Interchange: A scheduled possibility for passengers to change PT Vehicles, using two Stop Points (which may be identical) on two particular Journey Patterns.  
  corresponds to the Transmodel entity SERVICE JOURNEY PATTERN INTERCHANGE  
  (definition slightly adapted)

Advertised Journey Pattern Interchange: A property of a Journey Pattern Interchange indicating that the interchange is advertised.  
  corresponds to the attribute Advertised of SERVICE JOURNEY PATTERN INTERCHANGE

Guaranteed Journey Pattern Interchange: A property of a Journey Pattern Interchange indicating that the interchange is guaranteed.  
  corresponds to the attribute Guaranteed of SERVICE JOURNEY PATTERN INTERCHANGE

Journey Pattern Interchange Maximum Duration: The maximum allowed duration of a Journey Pattern Interchange.  
  corresponds to the attribute Maximum Duration of SERVICE JOURNEY PATTERN INTERCHANGE

Journey Pattern Interchange Standard Duration: The standard duration of a Journey Pattern Interchange.  
  corresponds to the attribute Standard Duration of SERVICE JOURNEY PATTERN INTERCHANGE

Journey Interchange: The scheduled possibility for transfer of passengers between two specific PT Vehicle Journeys, at the same or different Stop Points.  
  corresponds to the Transmodel entity SERVICE JOURNEY INTERCHANGE  
  (definition slightly adapted)
Advertised Journey Interchange: A property of a Journey Interchange indicating that the interchange is advertised.

Guaranteed Journey Interchange: A property of a Journey Interchange indicating that the interchange is guaranteed.

Journey Interchange Maximum Wait Time: The maximum wait time for the second vehicle in a Journey Interchange.

Interchange Status: The information about the actual status of a Journey Interchange on a specified Calendar Day.

TRIPS

Trip: The complete movement of a traveller from one place to another. A part of a Trip corresponding to the theoretical movement of a user on one and only one PT Vehicle.

Ride: A part of a Trip corresponding to the theoretical movement of a user on one and only one FT Vehicle.

PT Trip: A part of a Trip starting from the first boarding of a PT Vehicle to the last alighting from a PT Vehicle.

PT ORGANISATION

PT Company: A Public Transport operating company.

PT Company Name: The name of a PT Company.

Public Transport Mode: A part of a PT network characterised by means of transport (bus, tram, metro, train, ferry, etc.)

OTHER MAIN CONCEPTS USED BY PT

NDT Next Departure
This gives information about the next departure on a specified service.

NTI Next Departure Time
This attribute indicates the next time of departure of a ferry, or any public transport.

ATS Arrival Time
This indicates the time of the arrival of an individual vehicle on a detection zone.
PAS  Passage Time
This indicates the time taken by a vehicle in passing a point on the roadway.
(attribute)
This concept is related to the Transmodel entity ACTIVATION POINT

PRT  Presence Time
This indicates the time during which a vehicle activates a presence sensor.
(attribute)
This concept is related to the Transmodel entity ACTIVATION POINT

TTM  Travel Time
Travel Time is the time taken to travel between two specified points, by a specified route, including
any time taken by involuntary stops and delays.
(data object, phrase)
This information is related to the concept PT Run Time, described above.

ODM  Origin-destination Matrix
Flows of vehicles of passengers according to their origin and destination.
(data object)
This information is related to the concept Trip, described above.

DEC  Delays/Cancellations
Delays and cancellations are disruptions to traffic or PT services resulting in hold-ups, lateness or
unavailability of service.
(data object, desc. cat.)
This concept is related to the Transmodel entity JOURNEY MODIFICATION LOG ENTRY

SCN  Service Cancelled
A particular departure is cancelled.
(phrase)
This concept is related to the Transmodel entity JOURNEY MODIFICATION LOG ENTRY

DPN  No Departures
All services cancelled.
(phrase)
This concept is related to the Transmodel entity JOURNEY MODIFICATION LOG ENTRY

FER  Ferries/Trains
This item describes the availability of ferry and train services and provides information relating to
departures.
(data object)
This concept is related to several Transmodel entities

INF  Information
This item gives information about PT departures, availability of car parking, park-and-ride, and other
items being presented over an audio channel.
(desc. cat.)
This concept is related to several Transmodel entities
Appendix D

Sample TCP ECHO server

Private Sub ActionButton_Click()
If Not Socket1.Connected Then
    On Error Resume Next
    Hostname.Text = Trim$(Hostname.Text)
    IPAddress.Text = Trim$(IPAddress.Text)
    If Len(Hostname.Text) > 0 Then
        Socket1.Hostname = Hostname.Text
        IPAddress.Text = Socket1.HostAddress
    ElseIf Len(IPAddress.Text) > 0 Then
        Socket1.HostAddress = Trim$(IPAddress.Text)
        Hostname.Text = Socket1.Hostname
    Else
        MsgBox "No system name or address specified"
        Exit Sub
    End If
    ServiceName.Text = Trim$(ServiceName.Text)
    ServicePort.Text = Trim$(ServicePort.Text)
    If Len(ServiceName.Text) > 0 Then
        Socket1.RemoteService = ServiceName.Text
        ServicePort.Text = Trim$(Str$(Socket1.RemotePort))
    ElseIf Len(ServicePort) > 0 Then
        Socket1.RemotePort = Val(Trim$(ServicePort.Text))
        ServiceName.Text = Socket1.RemoteService
    Else
        MsgBox "No service name or port specified"
        Exit Sub
    End If
    Socket1.LocalPort = IPPORT_ANY
    Socket1.Action = SOCKET_CONNECT
    If Err <> 0 Then Exit Sub
    ActionButton.Enabled = False
    ActionButton.Default = False
Else
    Socket1.Action = SOCKET_CLOSE
    ActionButton.Caption = "Connect"
    ActionButton.Default = True
    Hostname.SetFocus
End If
Exit Sub
End Sub

Private Sub CancelButton_Click()
    Unload MainForm
End Sub

Private Sub ExitApp_Click()
    Unload MainForm
End Sub

Private Sub Form_Load()
    ' Initialize the socket control
    Socket1.AddressFamily = AF_INET
    Socket1.Binary = False
Socket1.Blocking = False
Socket1.BufferSize = 1024
Socket1.Protocol = IPPROTO_IP
Socket1.Type = SOCK_STREAM
Socket1.RemoteService = "echo"

Socket2.AddressFamily = AF_INET
Socket2.Binary = True
Socket2.Blocking = False
Socket2.HostAddress = INADDR_ANY
Socket2.Protocol = IPPROTO_IP
Socket2.Type = SOCK_STREAM
Socket2.LocalService = "echo"

' Initialize hostname combo box
On Error Resume Next
Socket1.HostFile = "HOSTS"
If Err = 0 Then
    Host$ = Socket1.GetFirstHost
    While Host$ <> 
        Hostname.AddItem Host$
        Host$ = Socket1.GetNextHost
    Wend
End If
Socket1.HostFile = ""

' Initialize the other controls
ActionButton.Caption = "Connect"
ActionButton.Default = True
ActionButton.Enabled = False
ServiceName.Text = Socket1.RemoteService
ServicePort.Text = Trim$(Str$(Socket1.RemotePort))
End Sub

Private Sub Form_Unload(Cancel As Integer)
    If Socket1.Connected Then Socket1.Action = SOCKET_CLOSE
    If Socket2.Listening Or Socket2.Connected Then Socket2.Action = SOCKET_CLOSE
End End Sub

Private Sub Hostname_Change()
    If Len(Hostname.Text) > 0 Or Len(IPAddress.Text) > 0 Then
        ActionButton.Enabled = True
    Else
        ActionButton.Enabled = False
    End If
End Sub

Private Sub Hostname_GotFocus()
    Hostname.SelStart = 0
    Hostname.SelLength = Len(Hostname.Text)
End Sub

Private Sub Hostname_KeyPress(KeyAscii As Integer)
    IPAddress.Text = ""
End Sub

183
Private Sub IPAddress_Change()
    If Len(Hostname.Text) > 0 Or Len(IPAddress.Text) > 0 Then
        ActionButton.Enabled = True
    Else
        ActionButton.Enabled = False
    End If
End Sub

Private Sub IPAddress_GotFocus()
    IPAddress.SelStart = 0
    IPAddress.SelLength = Len(IPAddress.Text)
End Sub

Private Sub IPAddress_KeyPress(KeyAscii As Integer)
    Hostname.Text = ""
End Sub

Private Sub ServerOption_Click()
    If Not Socket2.Connected Then
        ServerOption.Checked = Not ServerOption.Checked
        If ServerOption.Checked Then
            Socket2.Action = SOCKET_LISTEN
        Else
            Socket2.Action = SOCKET_CLOSE
        End If
    End If
End Sub

Private Sub ServiceName_GotFocus()
    ServiceName.SelStart = 0
    ServiceName.SelLength = Len(ServiceName.Text)
End Sub

Private Sub ServiceName_KeyPress(KeyAscii As Integer)
    ServicePort.Text = ""
End Sub

Private Sub ServicePort_GotFocus()
    ServicePort.SelStart = 0
    ServicePort.SelLength = Len(ServicePort.Text)
End Sub

Private Sub ServicePort_KeyPress(KeyAscii As Integer)
    ServiceName.Text = ""
    If KeyAscii > Asc("9") Then
        KeyAscii = 0
        Beep
    ElseIf KeyAscii >= Asc("0") And Len(ServicePort.Text) > 5 Then
        KeyAscii = 0
        Beep
    End If
End Sub

Private Sub Socket1_Close()
    Socket1.Action = SOCKET_CLOSE
    ActionButton.Caption = "Connect"
    ActionButton.Default = True
    Hostname.SetFocus
End Sub
Private Sub Socket1_Connect()
    MainForm.MousePointer = 0
    ActionButton.Caption = "Close"
    ActionButton.Enabled = True
    SocketInput.SetFocus
End Sub

Private Sub Socket1_Error(ErrCode As Integer, ErrMsg As String, Response As Integer)
    If Socket1.Action = SOCKET_CONNECT Then
        MsgBox ErrMsg, 0, "Connect"
        Response = SOCKET_ERRIGNORE
        Socket1.Action = SOCKET_CLOSE
        ActionButton.Caption = "Connect"
        ActionButton.Default = True
        ActionButton.Enabled = True
        Exit Sub
    End If
End Sub

Private Sub Socket1_Read(DataLength As Integer, IsUrgent As Integer)
    Socket1.RecvLen = DataLength
    SocketOutput.AddItem Socket1.RecvData
    SocketOutput.ListIndex = SocketOutput.ListCount - 1
    SocketOutput.Selected(SocketOutput.ListIndex) = False
End Sub

Private Sub Socket2_Accept(SocketId As Integer)
    sockets.Action = SOCKET_ACCEPT
End Sub

Private Sub Socket2_Close()
    Socket2.Action = SOCKET_CLOSE
    If ServerOption.Checked Then
        Socket2_REUSEADDRESS = True
        Socket2.Action = SOCKET_LISTEN
    End If
End Sub

Private Sub Socket2_Read(DataLength As Integer, IsUrgent As Integer)
    Socket2.RecvLen = DataLength
    Socket2.SendLen = DataLength
    Socket2.SendData = Socket2.RecvData
End Sub

Private Sub SocketInput_KeyPress(KeyAscii As Integer)
    If KeyAscii = 13 Then
        KeyAscii = 0
        Socket1.SendLen = Len(SocketInput.Text) + 2
        Socket1.SendData = SocketInput.Text & Chr(13) & Chr(10)
        SocketInput.Text = ""
    End If
End Sub

Private Sub Spin1_SpinDown()
    Dim Value As Integer
    Value = Val(ServicePort.Text) - 1
    If Value >= 0 Then
        Socket1.RemotePort = Value
        ServiceName.Text = Socket1.RemoteService
    End If
End Sub
ServicePort.Text = Trim(Str(Value))
Else
    Beep
End If
End Sub

Private Sub Spin1_SpinUp()
    Dim Value As Integer

    Value = Val(ServicePort.Text) + 1
    If Value > 0 Then
        Socket1.RemotePort = Value
        ServiceName.Text = Socket1.RemoteService
        ServicePort.Text = Trim(Str(Value))
    Else
        Beep
    End If
End Sub
Appendix E

The original published version of the protocol as created as a result of this thesis and tested within it. This version of the protocol has now been superseded by version 1.1.

JOURNEYWEB

<table>
<thead>
<tr>
<th>Title:</th>
<th>JW Query Interface - Standard Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status:</td>
<td>Implemented</td>
</tr>
<tr>
<td>Abstract:</td>
<td>This document describes the scope, principles and basic information on the query interface available to exchange data between journey planner systems</td>
</tr>
<tr>
<td>Primary Author:</td>
<td>Garrett Fingerle</td>
</tr>
<tr>
<td>Other Authors:</td>
<td>Jon Pheasant, Phil Smith, Mike Ness, Tony Lock, Roger Slevin, Roger Funnell, Paul Houghton, Steve Dawes and others</td>
</tr>
</tbody>
</table>

1 Document Control

Working Draft - Version 1.0 : Friday, 20 August 1999

2 Scope
This document includes working protocols for physical data string exchange for internet calls made to request "places", "exchanges", "journeys", but does not include work in progress regarding modifications and extensions to these protocol, such as "nearest points of travel", "routes", and eventually, "availability", "fares", "reservations", and "payment".

Note: schemas for XML used in this document are modelled after Microsoft examples. However, current w3 consortium drafts vary slightly from these according to early drafts available as of August, 1999, and later releases of this document will reflect those internationally agreed schemas and not just the recommendation as suggested by Microsoft.

3 Message : PLACE

Purpose: to determine a unique point of departure or arrival for journey that is a composite of trips.

Method: the home journey planner has a list of the unique community names from the national gazetteer. Linked to this list is the related URL inclusive of hostname, pathname, and filename of the journey planner responsible for that community. The home journey planner then selects a community name from the national gazetteer. Using the associated URL linked to that community in the National Gazetteer, the home journey planner then makes a remote request by this protocol for a list of "points" available for departure or arrival within that community

Note: a single request may contain more than one community, as long as the URL is identical. This occurrence might arise where both the origin and destination communities are in the same remote journey planner's database.

REQUEST FROM THE JOURNEY PLANNER TO THE JOURNEY ENGINE:

This section has two parts - the tags used and a series of samples

TAGS USED IN REQUEST: METHOD: POST

<table>
<thead>
<tr>
<th>Element Name</th>
<th>Parameters that need passing</th>
<th>Example</th>
<th>Mand</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROTOCOL</td>
<td>Protocol being used</td>
<td>PROTOCOL=JOURNEYWEB</td>
<td>Y</td>
<td>DONE</td>
</tr>
<tr>
<td>COMMAND</td>
<td>Type of command/request</td>
<td>COMMAND=PLACE</td>
<td>Y</td>
<td>DONE</td>
</tr>
<tr>
<td>PACKETID</td>
<td>Request identifier</td>
<td>PACKETID=</td>
<td>Y</td>
<td>DONE</td>
</tr>
</tbody>
</table>

188
The first four tags use the standard "keyword=value" notation. The community tag follows standard XML notation and is a "value" attached to the "packet" keyword. Order of the keywords is not important, but generally one would expect to see them in the order listed.

Sample Request

URL

A Uniform Resource Locator as found in the National Gazetteer will be of the form [hostname/path/filename].

In example: 152.78.96.181/scripts/JW_Bucks/JW_Bucks.cgi

This can be broken down into its components as necessary; usually this follows as hostname and path, the latter which includes the filename.

Hostname = "152.78.96.181"
Path = "scripts/JW_Bucks/JW_Bucks.cgi"

Headers

A sample header follows:

"Content-Type: application/x-www-form-urlencoded" Carriage Return Line Feed "Content-Length: " (length of the data packet) Carriage Return Line Feed

Data Packet

A sample data packet follows. It contains four keywords: protocol; command; packetid; and packet. The packet contains the data-string which has the listing of communities appended in serial.
Note, as these requests are sent in a "POST" method, it is not necessary to substitute a plus (+) for a space ( ) as it would if the data string were part of a "GET" method.

RESPONSE FROM JOURNEY ENGINE TO JOURNEY PLANNER

As before, a listing of the tags available for making a response from the journey engine to the journey planner are as follows:

SCHEMA and TAG ELEMENTS USED IN RESPONSE: METHOD: POST

```xml
<Schema xmlns="urn:schemas-microsoft-com:xml-data" xmlns:dt="urn:schemas-microsoft-com:datatypes">
    <ElementType name='name' content='textOnly'/>
    <ElementType name='placeid' content='textOnly'/>
    <ElementType name='type' content='textOnly'/>
    <ElementType name='place' content='eltOnly '>
        <element type='name'/>
        <element type='placeid'/>
        <element type='type'/>
    </ElementType>
    <AttributeType name='name' dt:type='string' required='yes'/>
    <ElementType name='community' content='eltOnly'>
        <attribute type='name'/>
        <element type='place'/>
    </ElementType>
    <ElementType name='packet' content='eltOnly'>
        <element type='community'/>
    </ElementType>
    <ElementType name='packetlen' content='textOnly'/>
    <ElementType name='packetid' content='textOnly'/>
    <ElementType name='protocol' content='textOnly'/>
</Schema>
```
Sample RESPONSE

<?xml version='1.0'?>
<PROTOCOL>JOURNEYWEB</PROTOCOL>
<PACKETID>industrial_id_here</PACKETID>
<PACKETLEN>value</PACKETLEN>
<PACKET>

    <COMMUNITY NAME="Amersham">
        <PLACE>
            <NAME>ACC Train^g Cen-Aldsht</NAME>
            <PLACEID>localidhere</PLACEID>
            <TYPE>timing point</TYPE>
        </PLACE>
        <PLACE>
            <NAME>Aldershot (Bus Stn)</NAME>
            <PLACEID>localidhere</PLACEID>
            <TYPE>timing point</TYPE>
        </PLACE>
    </COMMUNITY>
</PACKET>
4 MESSAGE : EXCHANGE

Purpose: to determine unique points of exchange available for changing between services within a journey

Method: select a locality name from the national gazetteer and it will link with the administrative area responsible for points within that locality. By requesting from the URL linked with that administrative area, it is then possible to request by this protocol a list of points available for departure or arrival within that locality
this request assumes that the home journey planner has the unique community name from the national gazetteer and its related URL inclusive of hostname, pathname, and filename.

Note: a single request may contain more than one community, as long as the URL is identical. This occurrence could be the case where both the origin and destination communities are in the same remote journey planner's database.

Status: implemented with revisions undergoing implementation and review

Call to remote journey planner for exchange points from place names:

List of Data TAGS

<table>
<thead>
<tr>
<th>Element Name</th>
<th>Parameters that need passing</th>
<th>Example</th>
<th>Mand</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROTOCOL=</td>
<td>Protocol being used</td>
<td>PROTOCOL=JOURNEYWEB</td>
<td>Y</td>
<td>DONE</td>
</tr>
<tr>
<td>COMMAND=</td>
<td>Command (request being made)</td>
<td>COMMAND=EXCHANGE</td>
<td>Y</td>
<td>DONE</td>
</tr>
<tr>
<td>PACKETID=</td>
<td>Packet ID</td>
<td>PACKETID=industrial_id_here</td>
<td>Y</td>
<td>DONE</td>
</tr>
<tr>
<td>PACKET=</td>
<td>Packet</td>
<td>PACKET=&lt;keyword&gt;value&lt;/keyword&gt;</td>
<td>Y</td>
<td>DONE</td>
</tr>
<tr>
<td>REQUESTID</td>
<td>Request ID</td>
<td>&lt;REQUESTID&gt;1&lt;/REQUESTID&gt;</td>
<td>N</td>
<td>DONE</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>First Community</td>
<td>&lt;COMMUNITY&gt;Addlestone&lt;/COMMUNITY&gt;</td>
<td>N</td>
<td>DONE</td>
</tr>
<tr>
<td>COMMUNITYID</td>
<td>First Community ID</td>
<td>&lt;COMMUNITYID&gt;E0000348&lt;/COMMUNITYID&gt;</td>
<td>N</td>
<td>DONE</td>
</tr>
<tr>
<td>PLACE</td>
<td>First place</td>
<td>&lt;PLACE&gt;ACC Train'g Cen-Aldsh&lt;/PLACE&gt;</td>
<td>N</td>
<td>DONE</td>
</tr>
<tr>
<td>PLACEID</td>
<td>First placeid</td>
<td>&lt;placeid&gt;localidhere&lt;/placeid&gt;</td>
<td>Y</td>
<td>DONE</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>Second Community</td>
<td>&lt;COMMUNITY&gt;Aldershot&lt;/COMMUNITY&gt;</td>
<td>N</td>
<td>DONE</td>
</tr>
<tr>
<td>COMMUNITYID</td>
<td>Second Community ID</td>
<td>&lt;COMMUNITYID&gt;Aldershot&lt;/COMMUNITYID&gt;</td>
<td>N</td>
<td>DONE</td>
</tr>
<tr>
<td>PLACE</td>
<td>Second place</td>
<td>&lt;PLACE&gt;Aldershot Bathing Pool&lt;/PLACE&gt;</td>
<td>N</td>
<td>DONE</td>
</tr>
<tr>
<td>PLACEID</td>
<td>Second placeid</td>
<td>&lt;placeid&gt;localidhere&lt;/placeid&gt;</td>
<td>N</td>
<td>DONE</td>
</tr>
</tbody>
</table>

Example of Request

PROTOCOL=JOURNEYWEB&COMMAND=EXCHANGE&PACKETID=industrial_id_here&PACKET=<REQUESTID>1</REQUESTID><PLACEID>357</PLACEID>
Returned results from remote journey planner:

SCHEMA and List of Data TAGS

<Schema xmlns="urn:schemas-microsoft-com:xml-data" xmlns:dt="urn:schemas-microsoft-com:datatypes">
  <AttributeType name='placeid' dt:type='string' required='yes'/>
  <ElementType name='localid' content='textOnly'/>
  <ElementType name='localname' content='textOnly'/>
  <ElementType name='localcommunity' content='textOnly'/>
  <ElementType name='exchangetime' content='textOnly'/>
  <ElementType name='exchangemode' content='textOnly'/>
  <ElementType name='exchangenatid' content='textOnly'/>
  <ElementType name='exchangesupid' content='textOnly'/>
  <ElementType name='exchangename' content='textOnly'/>
  <ElementType name='exchange' content='textOnly' />
      <attribute type='placeid'/>
      <element type='localid'/>
      <element type='localname'/>
      <element type='localcommunity'/>
      <element type='exchangetime'/>
      <element type='exchangemode'/>
      <element type='exchangenatid'/>
      <element type='exchangesupid'/>
      <element type='exchangename'/>
  </ElementType>
  <ElementType name='packet' content='textOnly'>
    <element type='exchange'/>
  </ElementType>
  <ElementType name='placeid' content='textOnly'/>
  <ElementType name='requestid' content='textOnly'/>
  <ElementType name='packetid' content='textOnly'/>
  <ElementType name='protocol' content='textOnly'/>
</Schema>
<table>
<thead>
<tr>
<th>Element Name</th>
<th>Attribute (Type)</th>
<th>Range (length)</th>
<th>Mand</th>
<th>Parameters that need passing</th>
<th>Example</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROTOCOL</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>Protocol being used</td>
<td>&lt;PROTOCOL&gt;value&lt;/PROTOCOL&gt;</td>
<td></td>
</tr>
<tr>
<td>PACKETID</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>Requester ID</td>
<td>&lt;PACKETID&gt;value&lt;/PACKETID&gt;</td>
<td></td>
</tr>
<tr>
<td>PACKET</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>Request data delimiter</td>
<td>&lt;PACKET&gt;value&lt;/PACKET&gt;</td>
<td></td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>Community</td>
<td>&lt;COMMUNITY&gt;Addlestone Area&lt;/COMMUNITY&gt;</td>
<td></td>
</tr>
<tr>
<td>EXCHANGE</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>Exchange point delimiter placed as sub</td>
<td>&lt;EXCHANGE PLACEID=357&gt;value&lt;/EXCHANGE&gt;</td>
<td></td>
</tr>
<tr>
<td>LOCALID</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>LOCALID</td>
<td>&lt;LOCALID&gt;749&lt;/LOCALID&gt;</td>
<td></td>
</tr>
<tr>
<td>LOCALNAME</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>LOCALNAME</td>
<td>&lt;LOCALNAME&gt;Bletchley Bus Station&lt;/LOCALNAME&gt;</td>
<td></td>
</tr>
<tr>
<td>LOCALCOMMUNITY</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>LOCALCOMMUNITY</td>
<td>&lt;LOCALCOMMUNITY&gt;E0039270&lt;/LOCALCOMMUNITY&gt;</td>
<td></td>
</tr>
<tr>
<td>EXCHANGETIME</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>EXCHANGETIME</td>
<td>&lt;EXCHANGETIME&gt;5&lt;/EXCHANGTIME&gt;</td>
<td></td>
</tr>
<tr>
<td>EXCHANGEMODE</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>EXCHANGEMODE</td>
<td>&lt;EXCHANGEMODE=WALK&lt;/EXCHANGEMODE&gt;</td>
<td></td>
</tr>
<tr>
<td>EXCHANGEIDATID</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>EXCHANGEIDATID</td>
<td>&lt;EXCHANGEIDATID=E0000285&lt;/EXCHANGEIDATID&gt;</td>
<td></td>
</tr>
<tr>
<td>EXCHANGESUPID</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>EXCHANGESUPID</td>
<td>&lt;EXCHANGESUPID=BLETCHLEY&lt;/EXCHANGESUPID&gt;</td>
<td></td>
</tr>
<tr>
<td>EXCHANGENAME</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>EXCHANGENAME</td>
<td>&lt;EXCHANGENAME&gt;BLETCHLEY&lt;/EXCHANGENAME&gt;</td>
<td></td>
</tr>
</tbody>
</table>

Example of Response

```xml
<PROTOCOL>JOURNEYWEB</PROTOCOL>
<PACKETID>TEST1</PACKETID>
<REQUESTID>1</REQUESTID>
<PLACEID>357</PLACEID>
<PACKET>
  <EXCHANGE PLACEID="357">
    <LOCALID>749</LOCALID>
    <LOCALNAME>Bletchley Bus Station</LOCALNAME>
    <LOCALCOMMUNITY>E0039270</LOCALCOMMUNITY>
    <EXCHANGETIME>5</EXCHANGETIME>
    <EXCHANGEMODE=WALK</EXCHANGEMODE>
  </EXCHANGE>
</PACKET>
```
<EXCHANGE PLACEID="357">
  <LOCALID>625</LOCALID>
  <LOCALNAME>M.Keynes Coachway</LOCALNAME>
  <LOCALCOMMUNITY>E0053461</LOCALCOMMUNITY>
  <EXCHANGETIME>5</EXCHANGETIME>
  <EXCHANGEMODE>WALK</EXCHANGEMODE>
  <EXCHANGENATID>R0001613</EXCHANGENATID>
  <EXCHANGESUPID>MKNSCEN</EXCHANGESUPID>
  <EXCHANGENAME>MILTON KEYNES CENTRAL</EXCHANGENAME>
</EXCHANGE>

<EXCHANGE PLACEID="357">
  <LOCALID>379</LOCALID>
  <LOCALNAME>M.Keynes Bus/Rail Stns</LOCALNAME>
  <LOCALCOMMUNITY>E0053461</LOCALCOMMUNITY>
  <EXCHANGETIME>5</EXCHANGETIME>
  <EXCHANGEMODE>WALK</EXCHANGEMODE>
  <EXCHANGENATID>NX000982</EXCHANGENATID>
  <EXCHANGESUPID>50124</EXCHANGESUPID>
  <EXCHANGENAME>MILTON KEYNES COACHWAY</EXCHANGENAME>
</EXCHANGE>

<EXCHANGE PLACEID="357">
  <LOCALID>2265</LOCALID>
  <LOCALNAME>Aylesbury Town Centre</LOCALNAME>
  <LOCALCOMMUNITY>E0000348</LOCALCOMMUNITY>
  <EXCHANGETIME>5</EXCHANGETIME>
  <EXCHANGEMODE>WALK</EXCHANGEMODE>
  <EXCHANGENATID>NX000982</EXCHANGENATID>
  <EXCHANGESUPID>56013</EXCHANGESUPID>
  <EXCHANGENAME>Aylesbury Coach Stop</EXCHANGENAME>
</EXCHANGE>
5 MESSAGE : SEARCH

SEARCHING FOR JOURNEYS FROM REMOTE JOURNEY PLANNER

Purpose: given a journey definition, to determine a selection of journeys that meet the given criteria

Method: a complete journey definition is sent to a home journey planner, which acts as an agent to search out and determine the necessary components and their locations; once the composite journeys have been found, the home journey planner compiles and returns a composite, unbiased selection of alternative journey solutions

Comm: the request is sent by http over port 80, and received also in the same manner.

Note: a single request may contain an embedded matrix of origin and destination pairs as long as the URL is identical

Status: parts of the protocol are implemented and tested whilst more recent additions are under implementation and testing

Call to remote journey planner for journey request from place names: optional flags are highlighted.

Within the post string, the order of fields within is not important but maintained for illustration, in so long as embedded tags maintain their integrity. Also, whether a tag appears in upper case or lower case or mixed is not important within the brackets. For instance, "<origin>" , "<ORIGIN>" , and "<Origin>" should all be treated as "<ORIGIN>". However, the case of the value within the field pair should be maintained. For instance, the value within the following example of "origin" should always return "Amersham": "<origin>Amersham<origin>".

197
<table>
<thead>
<tr>
<th>Marker Used</th>
<th>Mand.</th>
<th>Parameters that need passing</th>
<th>Example</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROTOCOL=</td>
<td>Y</td>
<td>Protocol being used</td>
<td>PROTOCOL=JOURNEYWEB</td>
<td>DONE</td>
</tr>
<tr>
<td>COMMAND=</td>
<td>Y</td>
<td>Command (request being made)</td>
<td>COMMAND=SEARCH</td>
<td>DONE</td>
</tr>
<tr>
<td>PACKETID=</td>
<td>Y</td>
<td>Requester ID</td>
<td>PACKETID=industrial id here</td>
<td>DONE</td>
</tr>
<tr>
<td>PACKET=</td>
<td>Y</td>
<td>List of journeys definitions in a matrix</td>
<td>PACKET=&lt;journey...</td>
<td>DONE</td>
</tr>
<tr>
<td>JOURNEY</td>
<td>Y</td>
<td>Journey definition delimiter</td>
<td>&lt;JOURNEY&gt;value&lt; /JOURNEY&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>REQUESTID</td>
<td>Y</td>
<td>Journey definition id</td>
<td>&lt;requestid&gt;anidhere&lt;/requestid&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>ORIGIN</td>
<td>Y</td>
<td>Origin</td>
<td>&lt;origin&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>DESTINATION</td>
<td>Y</td>
<td>Destination</td>
<td>&lt;destination&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>DATE</td>
<td>Y</td>
<td>Date</td>
<td>&lt;date&gt;25/03/1998&lt;/date&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>DATE</td>
<td>N</td>
<td>Date - long form</td>
<td>&lt;date format=dd/mm/yyyy&gt;25/03/1998&lt;/date&gt;</td>
<td>In progress</td>
</tr>
<tr>
<td>TIME</td>
<td>Y</td>
<td>Time</td>
<td>&lt;time&gt;08:00&lt;/time&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>TIME</td>
<td>N</td>
<td>Time - long form</td>
<td>&lt;time format=hh:mm&gt;08:00&lt;/time&gt;</td>
<td>In progress</td>
</tr>
<tr>
<td>RANGE</td>
<td>N</td>
<td>Range - Span of time</td>
<td>&lt;range format=interval&gt;01:00&lt;/range&gt;</td>
<td>In progress</td>
</tr>
<tr>
<td>RANGE</td>
<td>N</td>
<td>Range - number of journeys returned (next)</td>
<td>&lt;range format=sequence&gt;4&lt;/range&gt;</td>
<td>In progress</td>
</tr>
</tbody>
</table>

**NOTE:** to achieve a span of time that is 15 minutes before the traveller's departure time to 45 minutes after, it is necessary to subtract 15 minutes from TIME.
<table>
<thead>
<tr>
<th>RANGE</th>
<th>N</th>
<th>Range - number of journeys returned (previous)</th>
<th>&lt;range format=sequence&gt;-4&lt;/range&gt;</th>
<th>In progress</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARRDEP</td>
<td>Y</td>
<td>Departure or Arrival By</td>
<td>&lt;arrdep&gt;depart&lt;/arrdep&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>VIA</td>
<td>N</td>
<td>Via this list of places in this order or via these alternatives</td>
<td>&lt;via value=ordered&gt; &lt;community&gt;Amersham&lt;/community&gt; &lt;place&gt;ACC Train`g Cen-Aldsht&lt;/place&gt; &lt;placeid&gt;localidhere&lt;/placeid&gt;&lt;/via&gt; or &lt;via value=alternative&gt; &lt;community&gt;Amersham&lt;/community&gt; &lt;place&gt;ACC Train`g Cen-Aldsht&lt;/place&gt; &lt;placeid&gt;localidhere&lt;/placeid&gt;&lt;/via&gt;</td>
<td>Proposed</td>
</tr>
<tr>
<td>AVOID</td>
<td>N</td>
<td>Avoid this place or mode (NOTVIA?)</td>
<td>&lt;avoid&gt; &lt;community&gt;Amersham&lt;/community&gt; &lt;place&gt;ACC Train`g Cen-Aldsht&lt;/place&gt; &lt;placeid&gt;localidhere&lt;/placeid&gt;&lt;/avoid&gt; or &lt;avoid&gt;rail&lt;/avoid&gt;</td>
<td>In progress</td>
</tr>
</tbody>
</table>

Example of a POST request string with one journey

```
PROTOCOL=JOURNEYWEB&COMMAND=SEARCH&PACKETID=industrial_id_here&packet=<journey><requestid>prjwl</requestid><origin><community>Amerham</community><place>ACC Train\`g Cen-Aldsht</place><placeid>localidhere</placeid></origin><destination><community>Amerham</community><PLACE>Aldershot+Bathing+Pool</PLACE><placeid>localidhere</placeid></destination><date>25/03/1998</date><time>08:00</time><arrdep>Depart</arrdep></journey>
```

The integrity of the values in requestid tag and the packetid tag must be maintained. The values may be alphanumeric, and may include and underscore "_" and hyphen ",".

199
If no RANGE tag is included, the default is up to the implementers. However, it is necessary to specify what the capability of the implemented system is so as this default value can be included in the national gazetteer. Current default values include (wholeday, next journey).

Example of a POST request string with one journey with RANGE (interval) tag

```
PROTOCOL=JOURNEYWEB&COMMAND=SEARCH&PACKETID=industrial_id_here&packet=<journey><requestid>prjwl</requestid><origin><community>Amersham</community><place>ACC+Train"g+CeAldsh</place><placeid>localidhere</placeid></origin><destination><community>Amersham</community><PLACE>Aldershot+Bathing+Pool</PLACE><placeid>localidhere</placeid></destination><date>25/03/1998</date><time>08:00</time><arrdep>Depart</arrdep><range format=interval>12:00</range></journey>
```

This should be interpreted as requesting all journeys between these points, departing from (and including) 08:00 on the 25/March/1998 for the next twelve hours.

Example of a POST request string with one journey with RANGE (sequence) tag

```
PROTOCOL=JOURNEYWEB&COMMAND=SEARCH&PACKETID=industrial_id_here&packet=<journey><requestid>prjwl</requestid><origin><community>Amersham</community><place>ACC+Train"g+CeAldsh</place><placeid>localidhere</placeid></origin><destination><community>Amersham</community><PLACE>Aldershot+Bathing+Pool</PLACE><placeid>localidhere</placeid></destination><date>25/03/1998</date><time>08:00</time><arrdep>Depart</arrdep><range format=sequence>12</range></journey>
```

This should be interpreted as requesting the next 12 journeys between these points, departing from (and including) 08:00 on the 25/March/1998.

Example of a POST request string with two journeys

If a second journey were packaged in the same POST request, the string would look as follows:

```
PROTOCOL=JOURNEYWEB&COMMAND=SEARCH&PACKETID=industrial_id_here&packet=<journey><requestid>prjwl</requestid><origin><community>Amersham</community><place>ACC+Train"g+CeAldsh</place><placeid>localidhere</placeid></origin><destination><community>Amersham</community><PLACE>Aldershot+Bathing+Pool</PLACE><placeid>localidhere</placeid></destination><date>25/03/1998</date><time>08:00</time><arrdep>Depart</arrdep><range format=interval>12:00</range></journey>
```
Returned journeys from the remote journey planner:

```
<Schema xmlns="urn:schemas-microsoft-com:xml-data" xmlns:dt="urn:schemas-microsoft-com:datatypes">
  <ElementType name='community' content='textOnly'/>
  <ElementType name='place' content='textOnly'/>
  <ElementType name='placeid' content='textOnly'/>
  <ElementType name='bay' content='textOnly'/>
  <ElementType name='origin' content='eltOnly'>
    <element type='community'/>
    <element type='place'/>
    <element type='placeid'/>
    <element type='bay'/>
  </ElementType>
  <ElementType name='destination' content='eltOnly'>
    <element type='community'/>
    <element type='place'/>
    <element type='placeid'/>
    <element type='bay'/>
  </ElementType>
  <ElementType name='depart_time' content='textOnly'/>
  <ElementType name='depart_date' content='textOnly'/>
  <ElementType name='arrive_time' content='textOnly'/>
  <ElementType name='arrive_date' content='textOnly'/>
  <ElementType name='operator_name' content='textOnly'/>
  <ElementType name='service_number' content='textOnly'/>
  <ElementType name='mode' content='textOnly'/>
  <ElementType name='trip' content='eltOnly'>
    <element type='origin'/>
    <element type='depart_time'/>
    <element type='depart_date'/>
  </ElementType>
</Schema>
```
<element type='destination'/>
<element type='arrive_time'/>
<element type='arrive_date'/>
<element type='operator_name'/>
<element type='service_number'/>
<element type='mode'/>
</ElementType>
<ElementType name='journey' content='eltOnly'>
  <element type='trip'/>
</ElementType>
<AttributeType name='id' dt:type='string' required='yes'/>
<ElementType name='response' content='eltOnly'>
  <attribute type='id'/>
  <element type='journey'/>
</ElementType>
<ElementType name='packet' content='eltOnly'>
  <element type='response'/>
</ElementType>
<ElementType name='packetid' content='textOnly'/>
<ElementType name='packetlen' content='textOnly' require='no'/>
<ElementType name='protocol' content='textOnly'/>
</Schema>
<table>
<thead>
<tr>
<th>Element Name</th>
<th>Attribute (Type)</th>
<th>Range (length)</th>
<th>Mandatory</th>
<th>Description of parameters passing</th>
<th>Example</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROTOCOL</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>Protocol used</td>
<td>&lt;PROTOCOL&gt;journeyweb&lt;/PROTOCOL&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>PACKETID</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>Requester ID</td>
<td>&lt;PACKETID&gt;industrial_id_here&lt;/PACKETID&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>PACKET</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>Packet</td>
<td>&lt;packet&gt;journeys...&lt;/packet&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>PACKETLEN</td>
<td>String</td>
<td>Unlimited</td>
<td>N</td>
<td>Packet length</td>
<td>&lt;packetlen&gt;value&lt;/packetlen&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>RESPONSE</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>Response</td>
<td>&lt;response id=anidhere&gt;value&lt;/response&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>JOURNEY</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>Journey</td>
<td>&lt;JOURNEY&gt;value&lt;/JOURNEY&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>TRIP</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>Legs of journey</td>
<td>&lt;TRIP&gt;value&lt;/TRIP&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>DEPART_TIME</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>Depart time (always local)</td>
<td>&lt;DEPART_TIME format=HH:MM&gt;HH:MM&lt;/DEPART_TIME&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>&lt;DEPART_TIME format=HH:MM&gt;HH:MM&lt;/DEPART_TIME&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARRIVE_TIME</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>Arrive time (always local)</td>
<td>&lt;ARRIVE_TIME format=HH:MM&gt;08:00&lt;/ARRIVE_TIME&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEPART_DATE</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>Depart date (always local)</td>
<td>&lt;DEPART_DATE format=DD/MM/YYYY&gt;02/02/1999&lt;/DEPART_DATE&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARRIVE_DATE</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>Arrive date (always local)</td>
<td>&lt;ARRIVE_DATE format=DD/MM/YYYY&gt;02/02/1999&lt;/ARRIVE_DATE&gt;</td>
<td>DONE</td>
</tr>
</tbody>
</table>
| ORIGIN       | String          | Unlimited      | Y         | Depart place                       | <origin>
|              |                 |                |           |                                    | <community>Amersham</community>
|              |                 |                |           |                                    | <place>ACC Train'g Cen-Aldsht</place>
|              |                 |                |           |                                    | <PLACEID>localidhere</PLACEID>
|              |                 |                |           |                                    | <BAY>Bay 7</BAY>
|              |                 |                |           |                                    | <origin>                                                               | DONE   |
| COMMUNITY    | String          | Unlimited      | Y         | The communityid                    | <community>Amersham</community>                                         | DONE   |

203
<table>
<thead>
<tr>
<th>PLACE</th>
<th>String</th>
<th>Unlimited</th>
<th>Y</th>
<th>The name of the place</th>
<th>&lt;PLACE&gt;Aldershot Bathing Pool&lt;/PLACE&gt;</th>
<th>DONE</th>
</tr>
</thead>
<tbody>
<tr>
<td>PLACEID</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>The localid for the place</td>
<td>&lt;PLACEID&gt;localidhere&lt;/PLACEID&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>BAY</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>The name of the bay</td>
<td>&lt;BAY&gt;Bay 6&lt;/BAY&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>DESTINATION</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>Arrive place</td>
<td>&lt;destination&gt; &lt;community&gt;Amersham&lt;/community&gt; &lt;PLACE&gt;Aldershot Bathing Pool&lt;/PLACE&gt; &lt;PLACEID&gt;localidhere&lt;/PLACEID&gt; &lt;BAY&gt;Bay 6&lt;/BAY&gt; &lt;/destination&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>The communityid</td>
<td>&lt;community&gt;Amersham&lt;/community&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>PLACE</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>The place name</td>
<td>&lt;PLACE&gt;Aldershot Bathing Pool&lt;/PLACE&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>PLACEID</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>The localid for the place</td>
<td>&lt;PLACEID&gt;localidhere&lt;/PLACEID&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>BAY</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>The name of the bay</td>
<td>&lt;BAY&gt;Bay 6&lt;/BAY&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>OPERATOR NAME</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>Operators Name</td>
<td>&lt;OPERATOR NAME&gt;Arriva&lt;/OPERATOR NAME&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>SERVICE NUMBER</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>Service Number</td>
<td>&lt;SERVICE NUMBER&gt;734&lt;/SERVICE NUMBER&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>MODE</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>Mode of Transport</td>
<td>&lt;MODE&gt;bus&lt;/MODE&gt; &lt;MODE&gt;coach&lt;/MODE&gt; &lt;MODE&gt;rail&lt;/MODE&gt; &lt;MODE&gt;walk&lt;/MODE&gt; &lt;MODE&gt;cycle&lt;/MODE&gt; &lt;MODE&gt;taxi&lt;/MODE&gt; &lt;MODE&gt;air&lt;/MODE&gt; &lt;MODE&gt;ferry&lt;/MODE&gt; &lt;MODE&gt;tube/metro/lrt&lt;/MODE&gt; &lt;MODE&gt;commuter/RER&lt;/MODE&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>EXCHANGETIME</td>
<td>String</td>
<td>Unlimited</td>
<td>Y</td>
<td>Exchange Time</td>
<td>&lt;exchangetime&gt;00;05&lt;/exchangetime&gt;</td>
<td>DONE</td>
</tr>
<tr>
<td>EXCHANGETIME</td>
<td>String</td>
<td>Unlimited</td>
<td>N</td>
<td>Exchange Time</td>
<td>&lt;exchangetime format=&quot;hh:mm&quot;&gt;00:05&lt;/exchangetime&gt;</td>
<td>DONE</td>
</tr>
</tbody>
</table>
The exchange time is in practice considered the length of time to walk between platforms or bay, but is intended to provide a value for embarkment and disembarkment of the vehicle so as to more accurately portray an actual journey.

Example of a response with four options, some with multiple legs, journey string returned from the remote journey planner:

```xml
<PROTOCOL>journeyweb</PROTOCOL>
<PACKETID>industrial_id_here</PACKETID>
<PACKETLEN>value</PACKETLEN>
<PACKET>
  <RESPONSE ID="prjwl">
    <JOURNEY>
      <TRIP>
        <origin>
          <community>Amersham</community>
          <place>Amersham Station</place>
          <placeid>localidhere</placeid>
          <BAY>Hill Avenue Stop A</BAY>
        </origin>
        <DEPART_TIME>11:56</DEPART_TIME>
        <DEPART_DATE>21/10/1998</DEPART_DATE>
        <destination>
          <community>Amersham</community>
          <PLACE>Chesham Town Centre</PLACE>
          <placeid>localidhere</placeid>
          <BAY>Broadway Stop B</BAY>
        </destination>
        <ARRIVE_TIME>12:04</ARRIVE_TIME>
        <ARRIVE_DATE>21/10/1998</ARRIVE_DATE>
        <OPERATOR_NAME>Arriva</OPERATOR_NAME>
        <SERVICE_NUMBER>362</SERVICE_NUMBER>
        <MODE>bus</MODE>
      </TRIP>
      <TRIP>
        <origin>
          <community>Amersham</community>
          <place>Chesham Town Centre</place>
          <placeid>localidhere</placeid>
          <BAY>Broadway Stop B</BAY>
        </origin>
        <DEPART_TIME>12:09</DEPART_TIME>
        <DEPART_DATE>21/10/1998</DEPART_DATE>
        <destination>
          <community>Amersham</community>
          <PLACE>Chesham Town Centre</PLACE>
        </destination>
        <ARRIVE_TIME>12:10</ARRIVE_TIME>
        <ARRIVE_DATE>21/10/1998</ARRIVE_DATE>
        <OPERATOR_NAME>Arriva</OPERATOR_NAME>
        <SERVICE_NUMBER>362</SERVICE_NUMBER>
        <MODE>bus</MODE>
      </TRIP>
    </JOURNEY>
  </PACKET>
```
<TRIP>
<ARRIVE_TIME>12:06</ARRIVE_TIME>
<ARRIVE_DATE>21/10/1998</ARRIVE_DATE>
<OPERATOR_NAME>Chiltern Railways</OPERATOR_NAME>
<SERVICE_NUMBER>362</SERVICE_NUMBER>
<MODE>bus</MODE>
</TRIP>

<TRIP>
<origin>
<community>Great Missdn</community>
<place>Town Cent Railway Station</place>
<placeid>localidhere</placeid>
<BAY>Platform</BAY>
</origin>
<DEPART_TIME>12:08</DEPART_TIME>
<DEPART_DATE>21/10/1998</DEPART_DATE>
<destination>
<community>Great Missdn</community>
<PLACE>Town Cent</PLACE>
<placeid>localidhere</placeid>
<BAY></BAY>
</destination>
<ARRIVE_TIME>12:13</ARRIVE_TIME>
<ARRIVE_DATE>21/10/1998</ARRIVE_DATE>
<OPERATOR_NAME>Wycombe Bus</OPERATOR_NAME>
<SERVICE_NUMBER>372</SERVICE_NUMBER>
</TRIP>

<TRIP>
<origin>
<community>Great Missdn</community>
<place>Town Cent</place>
<placeid>localidhere</placeid>
<BAY></BAY>
</origin>
<DEPART_TIME>12:15</DEPART_TIME>
<DEPART_DATE>21/10/1998</DEPART_DATE>
<destination>
<community>Amerham</community>
<PLACE>Terriers Cross Roads</PLACE>
<placeid>localidhere</placeid>
<BAY>Crossroads</BAY>
</destination>
<ARRIVE_TIME>12:31</ARRIVE_TIME>
<ARRIVE_DATE>21/10/1998</ARRIVE_DATE>
<OPERATOR_NAME>Wycombe Bus</OPERATOR_NAME>
<SERVICE_NUMBER>372</SERVICE_NUMBER>
</TRIP>
<JOURNEY>
  <TRIP>
    <origin>
      <community>Amersham</community>
      <place>Amersham Station</place>
      <placeid>localidhere</placeid>
      <BAY></BAY>
    </origin>
    <DEPART_TIME>12:11</DEPART_TIME>
    <DEPART_DATE>21/10/1998</DEPART_DATE>
    <destination>
      <community>Amersham</community>
      <PLACE>Amersham Town Centre</PLACE>
      <placeid>localidhere</placeid>
      <BAY>Chiltern Ave Stop</BAY>
    </destination>
    <ARRIVE_TIME>12:06</ARRIVE_TIME>
    <ARRIVE_DATE>21/10/1998</ARRIVE_DATE>
    <MODE>walk</MODE>
  </TRIP>
  <TRIP>
    <origin>
      <community>Amersham</community>
      <place>Amersham Town Centre</place>
      <placeid>localidhere</placeid>
      <BAY>Chiltern Ave Stop</BAY>
    </origin>
    <DEPART_TIME>12:11</DEPART_TIME>
    <DEPART_DATE>21/10/1998</DEPART_DATE>
    <destination>
      <community>Amersham</community>
      <PLACE>Terriers Cross Roads</PLACE>
      <placeid>localidhere</placeid>
      <BAY>Crossroads</BAY>
    </destination>
    <ARRIVE_TIME>12:38</ARRIVE_TIME>
    <ARRIVE_DATE>21/10/1998</ARRIVE_DATE>
    <OPERATOR_NAME>Arriva</OPERATOR_NAME>
    <SERVICE_NUMBER>362</SERVICE_NUMBER>
    <MODE>bus</MODE>
  </TRIP>
</JOURNEY>
<exchangetime>00:05</exchangetime>
</JOURNEY>
<JOURNEY>
  <TRIP>
    <origin>
      <community>Amersham</community>
      <place>Amersham Station</place>
      <placeid>localidhere</placeid>
      <BAY>Hill Avenue Stop A</BAY>
    </origin>
    <DEPART_TIME>12:40</DEPART_TIME>
    <DEPART_DATE>21/10/1998</DEPART_DATE>
    <destination>
      <community>Amersham</community>
      <place>Chesham Town Centre</place>
      <placeid>localidhere</placeid>
      <BAY>Broadway Stop B</BAY>
    </destination>
    <ARRIVE_TIME>12:52</ARRIVE_TIME>
    <ARRIVE_DATE>21/10/1998</ARRIVE_DATE>
    <OPERATOR_NAME>Arriva</OPERATOR_NAME>
    <SERVICE_NUMBER>353</SERVICE_NUMBER>
    <MODE>bus</MODE>
  </TRIP>
  <TRIP>
    <origin>
      <community>Amersham</community>
      <place>Chesham Town Centre</place>
      <placeid>localidhere</placeid>
      <BAY>Broadway Stop B</BAY>
    </origin>
    <DEPART_TIME>12:54</DEPART_TIME>
    <DEPART_DATE>21/10/1998</DEPART_DATE>
    <destination>
      <community>Amersham</community>
      <PLACE>Chesham Town Centre</PLACE>
      <placeid>localidhere</placeid>
      <BAY>Broadway Stop A</BAY>
    </destination>
    <ARRIVE_TIME>12:59</ARRIVE_TIME>
    <ARRIVE_DATE>21/10/1998</ARRIVE_DATE>
    <MODE>walk</MODE>
  </TRIP>
  <TRIP>
    <origin>
      <community>Amersham</community>
      <place>Chesham Town Centre</place>
      <placeid>localidhere</placeid>
      <BAY>Broadway Stop A</BAY>
    </origin>
    <DEPART_TIME>12:54</DEPART_TIME>
    <DEPART_DATE>21/10/1998</DEPART_DATE>
    <destination>
      <community>Amersham</community>
      <PLACE>Chesham Town Centre</PLACE>
      <placeid>localidhere</placeid>
      <BAY>Broadway Stop A</BAY>
    </destination>
    <ARRIVE_TIME>12:59</ARRIVE_TIME>
    <ARRIVE_DATE>21/10/1998</ARRIVE_DATE>
    <MODE>walk</MODE>
  </TRIP>
</origin>

209
<community>Amersham</community>
<place>Chesham Town Centre</place>
<placeid>localidhere</placeid>
<BAY>Broadway Stop A</BAY>
</origin>
<DEPART_TIME>13:02</DEPART_TIME>
<DEPART_DATE>21/10/1998</DEPART_DATE>
<destination>
<community>Amersham</community>
<PLACE>Terriers Cross Roads</PLACE>
<placeid>localidhere</placeid>
<BAY>Crossroads</BAY>
</destination>
<ARRIVE_TIME>13:38</ARRIVE_TIME>
<ARRIVE_DATE>21/10/1998</ARRIVE_DATE>
<OPERATOR_NAME>Arriva</OPERATOR_NAME>
<SERVICE_NUMBER>362</SERVICE_NUMBER>
<MODE>bus</MODE>
</TRIP>
</JOURNEY>
</RESPONSE>

<!--JOURNEY would go here if there was more than one journey request in the packet-->

</RESPONSE>
</RESPONSE_ID="pqw2">
<//TRIP>
<//TRIP>
<//TRIP>
<//JOURNEY>
</RESPONSE>

<//JOURNEY>
</Journey>
</JOURNEY>
</JOURNEY>

<//Journey>
</JOURNEY>
</JOURNEY>

<//Journey>
</RESPONSE>

<//RESPONSE>

</packet>
Appendix F

Library of common XML functions

Function XML(tag As String, Value As String) As String
    XML = "<" & tag & ""> & Value & "</" & tag & ">
End Function

Function mXML(tag As String, modifier As String, Value As String) As String
    mXML = "<" & tag & Chr(32) & modifier & ">" & Value & "</" & tag & ">
End Function

Public Function retrieve_field(name_of_field As String, source_str As String) As String
' this function retrieves the string value of a field from a POST command
' The source string is then returned minus the value
    Dim Pos_Start As Long, Pos_End As Long, name_of_field_len As Long
    Pos_Start = FindStr(source_str, name_of_field) 'determines if the string exists
    If Pos_Start = 0 Then 'if the string does not exist
        retrieve_field = "Error-7 : no field" 'no named field
        Exit Function
    Else
        name_of_field_len = Len(name_of_field)
        Pos_End = InStr(Pos_Start + name_of_field_len, source_str, 
        If Pos_End = 0 Then 'must be last field use upto end of source_str
            source_str = source_str & 
        Pos_End = Len(source_str)
        End If 'assigns the value of the response named field to the local variable
        retrieve_field = Mid(source_str, Pos_Start + name_of_field_len + 1, Pos_End - Pos_Start - name_of_field_len - 1) 'chops out the portion of the string which contains the above response named field
        source_str = Left(source_str, Pos_Start - 1) & Right(source_str, Len(source_str) - Pos_End)
    End If
End Function

Public Function retrieve_value(name_of_tag As String, source_str As String, modifiers As String) As String
' this function retrieves the string within a tag and a string containing the modifiers
' The source string is then returned minus the value
    Dim Pos_Start As Long, Pos_End As Long, ErrorPos As Long
    Pos_Start = FindStr(source_str, name_of_tag & "">") 'determines if the string exists
    If Pos_Start = 0 Then 'if the string does not exist
        Pos_Start = FindStr(source_str, "" & name_of_tag & "">") 'test for string with modifiers
        If Pos_Start = 0 Then 'if no modifiers and
            retrieve_value = "Error-7 : no tag" 'no named tag with or without modifiers & Pos_Start & source_str
        source_str
    211
Exit Function
Else
    Pos_End = InStr(Pos_Start + Len(name_of_tag) + 1, source_str, ">") ' Find the bracket of the
    identifier
    If Pos_End = 0 Then
        retrieve_value = "Error - Incorrect or incomplete string format in protocol"
        Else 'get modifiers - assigns the keys and values of the named field to the modifiers
            modifiers = Mid(source_str, Pos_Start + Len(name_of_tag) + 2, Pos_End - (Pos_Start +
                Len(name_of_tag) + 2))
            'chop out the portion of the string which contains the modifiers
            source_str = Left(source_str, Pos_Start + Len(name_of_tag)) & Right(source_str,
                Len(source_str) - Pos_End + 1)
            End If
    ErrorPos = InStr(Pos_Start + Len(name_of_tag), UCase(source_str), "</" &
        UCase(name_of_tag) & ">")
    If ErrorPos = 0 Then 'assign an error message if the message is incomplete or incorrectly
        formatted
        retrieve_value = "Error - Incorrect or incomplete string format in protocol - No closing tag -
            modifiers" & Pos_Start
        Else 'assign the value of the response named field to the local variable
            retrieve_value = Mid(source_str, Pos_Start + Len(name_of_tag) + 2, ErrorPos - Pos_Start -
                Len(name_of_tag) - 2)
            'chops out the portion of the string which contains the above response named
            field
            source_str = Left(source_str, Pos_Start - 1) & Right(source_str, Len(source_str) - ErrorPos -
                Len(name_of_tag) - 2)
            End If
    End If
Else ' String exists without modifiers
    Pos_End = InStr(Pos_Start + Len(name_of_tag), UCase(source_str), ">")
    If Pos_End = 0 Then 'assigns an error message that no closing tag was found
        retrieve_value = "Error - Incorrect or incomplete string format in protocol - No closing tag - no
            modifiers" & Pos_Start
        Else 'assigns the value of the response named field to the local variable
            retrieve_value = Mid(source_str, Pos_Start + Len(name_of_tag) + 2, Pos_End - Pos_Start -
                Len(name_of_tag) - 2)
            'chops out the portion of the string which contains the above response named field
            source_str = Left(source_str, Pos_Start - 1) & Right(source_str, Len(source_str) - Pos_End -
                Len(name_of_tag) - 2)
            End If
    End If
End If
End Function

Public Function FindStr(MainStr As String, SubStr As String) As Long
Dim i As Long, j As Long
For i = 1 To Len(MainStr) - Len(SubStr) + 1
    j = 0
    Do While UCase(Mid(MainStr, i + j, 1)) = UCase(Mid(SubStr, j + 1, 1))
        j = j + 1
    Loop
    If j = Len(SubStr) Then
        FindStr = i
        Exit Function
    End If
Next i
FindStr = 0
Sample code for Passive Engine

Private Sub passiveengine_respond()

    Dim qProtocol As String, qCommand As String, qPacketID As String, qPacket As String, qTime As String
    Dim HostName As String, Path As String, sData As String, Headers As String
    Dim buf As String
    Dim i As Integer
    Dim qstr As String ' The CGI Querystring
    Dim modifier As String
    Dim packet As String
    Dim sURLData As String
    Dim binread As Variant
    Dim bytecount As Long

    On Error GoTo OnPostError ' Errors are handled here

    bytecount = Request.TotalBytes
    binread = Request.BinaryRead(bytecount)

    For i = 0 To bytecount - 1
        sURLData = sURLData & Chr$(binread(i))
    Next i

    qProtocol = retrieve_field("PROTOCOL", sURLData)
    qCommand = retrieve_field("COMMAND", sURLData)
    qPacketID = retrieve_field("PACKETID", sURLData)
    qPacket = retrieve_field("PACKET", sURLData)

    Select Case qProtocol

    Case "JOURNEYWEB"

        Select Case qCommand

        Case "PLACE"

            ' The purpose of the place command is to take a locality from the national gazetteer and return a listing of bus stops in the current version of development, and eventually, any point of departure including an address or postcode.

            buf = XML("PROTOCOL", qProtocol)
            buf = buf & XML("PACKETID", qPacketID)

            placeindex = 0
            Do
                placeindex = placeindex + 1
                place(placeindex).Name = retrieve_value("COMMUNITY", qPacket, vbNull)
            Loop Until Left(place(placeindex).Name, 5) = "Error"

            i = 0
            For i = 1 To placeindex - 1

                qstr = "StopsInBucks"
                Set db = DBEngine.Workspaces(0).OpenDatabase("c:\LiveData\bucks.mdb")
                Set qd = db.QueryDefs(qstr)

            Next i

    Case Else

        Error

    End Select

End Sub

213
qd.Parameters!pMatch = place(i).Name  
Set ds = qd.OpenRecordset(dbOpenDynaset)  
Do Until ds.EOF  
place(i).slist = place(i).slist & XML("PLACE", XML("NAME", ds("place_name") & 
"--" & ds("bay_name")) & XML("PLACEID", ds("Placenumber"))) & vbCrLf & vbTab  
ds.MoveNext  
Loop  
ds.Close  
qd.Close  
db.Close  

packet = packet & mXML("COMMUNITY", "NAME=" & Chr(34) & place(i).Name & 
Chr(34), vbCrLf & place(i).slist) & vbCrLf  
Next  
buf = buf & XML("PACKET", packet)  

Case "EXCHANGE"  
'The purpose of this command is to return the list of exchange points for each point of 
transport.  

buf = buf & XML("PROTOCOL", qProtocol)  
buf = buf & XML("PACKETID", qPacketID)  
buf = buf & qPacket  
placeindex = 0  
Do  
placeindex = placeindex + 1  
'place(placeindex).place = retrieve_value("PLACE", qPacket, vbNull)  
'place(placeindex).Name = retrieve_value("NAME", place(placeindex).place, vbNull)  
place(placeindex).placeid = retrieve_value("PLACEID", qPacket, vbNull)  
Loop Until Left(place(placeindex).placeid, 5) = "Error"  
i = 0  
For i = 1 To placeindex - 1  
qstr = "ExchangeFromLocality"  
Set db = DBEngine.Workspaces(0).OpenDatabase("c:\LiveData\bucks.mdb")  
Set qd = db.QueryDefs(qstr)  
qd.Parameters!pMatch = place(i).placeid  
Set ds = qd.OpenRecordset(dbOpenDynaset)  
Do Until ds.EOF  
place(i).slist = place(i).slist & mXML("EXCHANGE", "PLACEID=" & Chr(34) & place(i).placeid & Chr(34), vbCrLf & place(i).slist) & 
XML("LOCALID", ds("local_gazetteer.Placenumber")) & 
XML("LOCALNAME", ds("place_name")) & 
XML("LOCALCOMMUNITY", ds("gaz_id")) & 
XML("EXCHANGETIME", ds("exchangetime")) & 
XML("EXCHANGEMODE", ds("exchangemode")) & 
XML("EXCHANGENATID", ds("expt_id")) & 
XML("EXCHANGESUPID", ds("supplierid")) & 
XML("EXCHANGENAME", ds("expt_name")))  
'EXCHANGEMODE can have values of "BIKE", "WALK",  
or "TAXI"  
'EXCHANGETIME is defined in minutes  
'EXCHANGE_EGRESS time to get off the train or bus - only  
an issue for long trains like EuroStar
EXCHANGE_ACCESS  time to get on the train or bus - only an issue for long trains like EuroStar or queues such as in Greece
EXCHANGE_TYPE  for instance, definition specifying whether to add a % tolerance for older people, baggage handling, and so forth.

ds.MoveNext
Loop
ds.Close
qd.Close
db.Close

packet = packet & place(i).slist

Next
buf = buf & XML("PACKET", packet)

Case "SEARCH"

' The purpose of the search command is to receive the journey-web protocol and respond with a selection of journeys. In this case however, I am simply forwarding the command to the search engine after modifying the origin and destination fields if appropriate as search engines respond more politely to numbers than ambiguous strings of characters.

HostName = "152.78.96.181"
Path = "scripts/JW_Bucks/JW_Bucks.cgi"
Path = ""
Data = "protocol=JOURNEYWEB&PACKETID=TheJourneyID&COMMAND=SEARCH&packet=<journey><requestid>A</requestid><origin><placeid>" & UCase(qorigin) & "</placeid></origin><destination><PLACEid>" & UCase(qdestin) & "</PLACEid></destination><date>" & qdate & "</date><time>" & qTime & "</time><arrdep>depart</arrdep><range format=interval>6:00</range></journey>" & vbCrLf
sData = "PROTOCOL= & qProtocol & "&COMMAND=SEARCH&PACKETID= & qPacketID & "&PACKET= & qPacket & vbCrLf
Headers = "Content-Type: application/x-www-form-urlencoded" & vbCrLf & "Content-Length: " & Len(sData) & vbCrLf & vbCrLf

buf = HTTP_Client_Post(HostName, Path, sData, Len(sData), Headers, Len(Headers))

Case Else
buf = "Command " & qCommand & " not (yet) supported"
End Select
Case Else
buf = "Not a JourneyWeb Protocol"
End Select

'Write a reply to the user
With Response
  .Write "<html>"
  .Write "<body>"
  .Write "<h1>JourneyWeb Server Back End - Bucks</font></h1>"
  .Write "<p>" & Now & "</p>"
  .Write "<p>" & "152.78.96.181" & "</p>"
  .Write "<p>" & buf & "</p>"
End Response

215
.Write "</body>"
.Write "</html>
End With

DoPostFinish:       ' Can come here via error,
                   ' State of ds & qd unknown
    On Error Resume Next   ' Make sure ds and qd are closed
    ds.Close           ' else db.Close will fail and you lose
    qd.Close
    db.Close
    Exit Sub

' ------------------------
' Exception Handler
' ------------------------
',

OnError:
' If Err >= CGI_ERR_START Then Error Err ' Resignal if a CGI.BAS error
    buf = buf & "<H2>There was a problem:</H2>"
    buf = buf & "VB reports: <CODE>" & Error & " (error #" & Err & ")</CODE><H3>Best Guess:""
    buf = buf & "</H3>"
    Response.Write (buf)
    Resume DoPostFinish

End Sub
Appendix G

Testing Software Code

CGIMain.bas

Option Explicit

' Declare variables for database
Dim db As Database
Dim qd As QueryDef
Dim ds As Recordset

Private Type recordarray
    Name As String ' place
gazid As String
End Type

Private rid(1 To 3000) As recordarray ' defines a bounded array

' Declare variables for getting the path
Public sSelector As String

Sub CGI_Main()
    sSelector = UCase$(Mid$(CGI_PathInfo, 2)) ' Remove leading "/
    SendStart ("Southampton University")
    Select Case UCase$(CGI_RequestMethod)
        Case "GET":
            DoGet
        Case "POST":
            DoPost
        Case Else:
            Send ("<H2>Cannot do " & CGI_RequestMethod & " method</H2>")
    End Select
    SendStop
End Sub

Sub DoGet()
    Dim LinkStart As String
    LinkStart = "<A HREF=""http://" & CGI_ServerName & CGI_ScriptName
    Select Case sSelector
        ' This presents the opening page
        Case "" ' No "selector"
            Call MainTest("Automated Testing Software")
        Case "TOC"

217
Case Else:

    Send ("<H2>Bad GET selector " & sSelector & "</H2>")

End Select

End Sub

Sub SendErrorMessage(sMessage As String, sError As String)

    Send ("<p>" & sMessage & sError & " : Please contact customer service (between 1000 to 1230 and 13:00 to 1530) at 023 80 594660 (Mr. Fingerle at Southampton)</p>")

End Sub

Sub DoPost()

' This set of variables is necessary to collect the values from the BROWSER

Dim LinkStart As String

LinkStart = "<A HREF=" & CGI_ServerName & CGI_ScriptName

On Error GoTo OnPostError  ' Errors are handled here

Select Case sSelector

    Case ""

' No "selector", list choices

    Case "TEST"

      Call ProcessTest(GetCgiValue("testname"), _
                        GetCgiValue("testvalue"), _
                        GetCgiValue("quantity"), _
                        GetCgiValue("command"), _
                        "Results of Test", GetCgiValue("origin"), _
                        GetCgiValue("destination"), _
                        GetCgiValue("testdate"), _
                        GetCgiValue("testtime"), _
                        GetCgiValue("testda"), _
                        GetCgiValue("testrange")

    Case "TESTBATCH"

      Call Processbatch(GetCgiValue("testname"), _
                         GetCgiValue("testvalue"), _
                         GetCgiValue("quantity"))

    Case Else:

      Send ("<H2>Unknown POST selector " & sSelector & "</H2>")

End Select

DoPostFinish:  ' Can come here via error,

    On Error Resume Next  ' State of ds & qd unknown
    ds.Close  ' Make sure ds and qd are closed
    qd.Close
    db.Close
    Exit Sub

End Sub
' Exception Handler
' OnPostError:
'    If Err >= CGI_ERR_START Then Error Err  ' Resignal if a CGI.BAS error
        Send ("<H2>There was a problem:</H2>")
        Send ("<VB reports: <CODE>" & Error$ & " (error #" & Err & ")</CODE><br><b>Best Guess:"))
    Select Case sSelector
        Case ":
            SendStart ("Error in creating home page")
            Send ("Nothing Here")
            SendFooter
        Case Else:
            'SendStart
            If Err = 5 Then
                Send ("Please make certain that you have selected an item from the list."</b>)
            Else
                Send ("Programmer error: Unknown selector in POST exception handler."</b>)
            End If
            SendFooter
        End Select
    Send ("</H3>")
    Resume DoPostFinish
End Sub
Sub Inter_Main()
    MsgBox "This is a Windows CGI program"
End Sub
Sub OptionList(FieldName As String, Tbl As String, Col As String)
    Send ("Select " & FieldName & ": <SELECT NAME=" & Chr(34) & FieldName & Chr(34) & 
" size = 4>"
    Set ds = db.OpenRecordset(Tbl, dbOpenDynaset)
    Do Until ds.EOF
        Send ("<OPTION>" & ds(Col))
        ds.MoveNext
    Loop
    ds.Close
    Send ("</SELECT>"
End Sub
modActions.bas

Option Explicit

' Declare variables for database
Dim db As Database
Dim qd As QueryDef
Dim ds As Recordset

Public Sub MainTest(sMessage As String)
    Call SendTop(sMessage, CGI_ServerName, CGI_ScriptName)
    ' Send opening page
    Send ("<hr><pre>"
    Send ("<form method=""POST"" ACTION=""http://" &
    CGI_ServerName & CGI_ScriptName & "/TEST"">"
    Send ("<strong>Name of Test </strong>:  <INPUT TYPE=""TEXT"" NAME=""testname"" VALUE="">"
    Send ("<strong>Test Value </strong>:  <INPUT TYPE=""TEXT"" NAME=""testvalue"" VALUE="">"
    Send ("<strong>Population Size </strong>:  <INPUT TYPE=""TEXT"" NAME=""quantity"" VALUE="">"
    Send ("<strong>Purpose </strong>:  <INPUT TYPE=""TEXT"" NAME=""purpose"" VALUE="">"
    Send ("Select Command:  <SELECT NAME=""command"">"
    Send (<OPTION>PLACE"
    Send (<OPTION>EXCHANGE"
    Send (<OPTION>SEARCH"
    Send (<SELECT>"
    Send ("<strong>Origin Value </strong>:  <INPUT TYPE=""TEXT"" NAME=""origin"" VALUE="">"
    Send ("<strong>Destination Value</strong>:  <INPUT TYPE=""TEXT"" NAME=""destination"" VALUE="">"
    Send ("<strong>Date Value </strong>:  <INPUT TYPE=""TEXT"" NAME=""testdate"" VALUE="">"
    Send ("<strong>Time Value </strong>:  <INPUT TYPE=""TEXT"" NAME=""testtime"" VALUE="">"
    Send ("<strong>Depart Value </strong>:  <INPUT TYPE=""TEXT"" NAME=""testda"" VALUE="">"
    Send ("<INPUT TYPE=SUBMIT VALUE=""Start Test"">"
    Send ("</form>"
    Send ("</form></pre>"
End Sub

Public Sub Processbatch(TestName As String, _
    TestValue As String, _
    TestQuantity As String, _
    Optional sMessage As String)
Call SendTop(sMessage, CGI_ServerName, CGI_ScriptName)
Send (TestName & "<BR>")
Send (TestValue & "<BR>")
Send (TestQuantity & "<BR>")

End Sub

Public Sub ProcessTest(TestName As String, _
TestValue As String, _
TestQuantity As String, _
TestCommand As String, _
sMessage As String, _
Optional TestOrigin As String, _
Optional TestDestination As String, _
Optional TestDate As String, _
Optional TestTime As String, _
Optional TestDepArr As String, _
Optional TestRange As String)

Call SendTop(sMessage, CGI_ServerName, CGI_ScriptName)
Send (TestName & "<BR>")
Send (TestValue & "<BR>")
Send (TestQuantity & "<BR>")
Send (TestCommand & "<BR>")
Send (TestOrigin & "<BR>")
Send (TestDestination & "<BR>")
Send (TestDate & "<BR>")
Send (TestTime & "<BR>")
Send (TestDepArr & "<BR>")

If TestCommand = "PLACE" Then
  Call SendCommand("152.78.96.172/scripts/dual/bucks/bucksswap.asp", _
  TestCommand, TestName, TestValue, TestQuantity)
End If

If TestCommand = "EXCHANGE" Then
  Call SendCommand("152.78.96.172/scripts/dual/bucks/bucksswap.asp", _
  TestCommand, TestName, TestValue, TestQuantity)
End If

If TestCommand = "SEARCH" Then
  'Send ("SEARCH")
  Call SendCommand("152.78.96.172/scripts/dual/bucks/bucksswap.asp", _
  TestCommand, TestName, TestValue, TestQuantity, TestOrigin, _
  TestDestination, _
  TestDate, TestTime, TestDepArr, TestValue)
End If

End Sub

Public Sub SendCommand(url As String, _
TestCommand As String, _
TestName As String, _
TestValue As String, _
TestQuantity As String, _
Optional TestOrigin As String, _
Optional TestDestination As String, _
Optional TestDate As String, _
Optional TestTime As String, _
Optional TestDepArr As String, _
Optional TestRange As String)

Call SendTop(sMessage, CGI_ServerName, CGI_ScriptName)
Send (TestName & "<BR>")
Send (TestValue & "<BR>")
Send (TestQuantity & "<BR>")
Call SendTop(sMessage, CGI_ServerName, CGI_ScriptName)
Send (TestName & "<BR>")
Send (TestValue & "<BR>")
Send (TestQuantity & "<BR>")
End Sub
Optional TestTime As String, _
Optional TestDepArr As String, _
Optional TestRange As String)

Dim HostName As String
Dim Path As String
Dim Data As String
Dim Headers As String
Dim i, j, k, m As Long
Dim buf As String
Dim totalbuf As String
Dim StartTime
Dim MeanTimeEven
Dim MeanTimeOdd
If CInt(TestQuantity) > 2000 Then
  i = 1
ElseIf CInt(TestQuantity) < 1 Then
  i = 1
End If

HostName = Left(url, InStr(url, "/") - 1)  ' "152.78.96.181"
Path = Right(url, Len(url) - Len(HostName) - 1)
' "scripts/JW_Bucks/JW_Bucks.cgi"

Send (HostName)
Send (Path)

If TestCommand = "PLACE" Then
  Data = "PROTOCOL=JOURNEYWEB&COMMAND=PLACE&PACKETID=" _
    & Format(Now, "hh:mm:ss") & 
    & "&REQUESTID=" _
    & TestName & 
"</REQUESTID><COMMUNITY>" & TestValue & 
"</COMMUNITY>" & vbCrLf
End If

If TestCommand = "EXCHANGE" Then
  Data = "PROTOCOL=JOURNEYWEB&COMMAND=EXCHANGE&PACKETID=" _
    & Name & 
"&REQUESTID=" _
    & "&PLACEID=" & TestValue & 
"</PLACEID>" & vbCrLf
End If

If TestCommand = "SEARCH" Then
  m = 1
  Do While m < 11
    If m = 1 Then
      TestOrigin = "R0000006" 'Aberdeen Close
      TestDestination = "R0000007" 'Achanalt
    End If
    If m = 2 Then
      TestOrigin = "R0000006" 'Aberdeen to Wales
      TestDestination = "R0000143" 'Bangor
    End If
    If m = 3 Then
      TestOrigin = "R0000006" 'Aberdeen to West England
      TestDestination = "R0000843" 'Exeter Central
    End If
    If m = 4 Then
      TestOrigin = "R0000844" 'Ex. St Davids High Density
      TestDestination = 
      TestDestination = 
      Close
TestDestination = "R0000843" 'Ex Central
End If
If m = 5 Then
    TestOrigin = "R0000843" 'Ex Central Round the Severn
End If
If m = 6 Then
    TestOrigin = "R0002284" 'Swansea Round Wales
    TestDestination = "R00000143" 'Bangor
End If
If m = 7 Then
    TestOrigin = "R0002158" 'Southampton South to North
    TestDestination = "R0000006" 'Aberdeen
End If
If m = 8 Then
    TestOrigin = "R0000143" 'Bangor North Wales West to East
    TestDestination = "R0001925" 'Ramsgate
End If
If m = 9 Then
    TestOrigin = "R0001925" 'Rainham nr. Ramsgate East to North
    TestDestination = "R0000006" 'Aberdeen
End If
If m = 10 Then
    TestOrigin = "R0000843" 'Eve Central South West to East
    TestDestination = "R0001925" 'Rainham nr. Ramsgate
End If
'If k > 20 Then k = 20
'For k = 1 To 2 'TestQuantity
Data = "PROTOCOL=JOURNEYWEB&COMMAND=SEARCH&PACKETID=seqI" & "2" & 
    "&IIdpairI" & m & "&PACKET=<JOURNEY><REQUESTID>1</REQUESTID>" _
    & "<ORIGIN><COMMUNITY>RAIL STATION</COMMUNITY><PLACE>SOMEPLACE</PLACE><PLACEID>" & TestOrigin & 
    "</PLACEID></ORIGIN>" _
    & "<DESTINATION><COMMUNITY>RAIL STATION</COMMUNITY><PLACE>SOMEPLACE</PLACE><PLACEID>" & TestDestination & 
    "</PLACEID></DESTINATION><DATE>07/08/2000</DATE><TIME>12:00</TIME><ARR
    DREP>depart</ARRDREP><RANGE FORMAT=SEQUENCE>2</RANGE></JOURNEY>
"Send (Data) & "<br>"
Headers = "Content-Type: application/x-www-form-urlencoded" & vbCrLf & "Content-Length: " & Len(Data) & vbCrLf & vbCrLf
For i = 1 To TestQuantity
    j = i
    StartTime = Timer
    buf = HTTP_Client_Post(HostName, Path, Data, Len(Data), Headers, Len(Headers))
    Send ("TestIndex: " & Format(i, "#000.0") & " : " & Format(Timer - StartTime, "####0.000") & " seconds; ")
    If j Mod 2 = 0 Then
        MeanTimeEven = MeanTimeEven + (Timer - StartTime)
    End If
Next i
Send ("Mean Time for PIII-500 is : " & Format(MeanTimeEven / (j / 2), "0.000") & "<br>")
Else
    MeanTimeOdd = MeanTimeOdd + (Timer - StartTime)
    Send ("Mean Time for PII-450 is : " & Format(MeanTimeOdd / ((j + 1) / 2), "0.000") & "<br>")
End If
If i > TestQuantity - 2 Then
    totalbuf = totalbuf & buf
End If
Next i
Send ("<hr>")
If m = 10 Then
    Send (totalbuf)
End If
'Next k
m = m + 1
Loop
Else
    Send (Data) & "<hr>"
    Headers = "Content-Type:  application/x-www-form-urlencoded" & vbCrLf & "Content-Length:  " & Len(Data) & vbCrLf & vbCrLf
    For i = 1 To TestQuantity
        j = i
        StartTime = Timer
        buf = HTTP_Client_Post(HostName, Path, Data, Len(Data), Headers, Len(Headers))
        Send ("TestIndex:  " & Format(i, "#000.0") & " : " & Format(Timer - StartTime, "#####0.000") & " seconds; ")
        If j Mod 2 = 0 Then
            MeanTimeEven = MeanTimeEven + (Timer - StartTime)
            Send ("Mean Time for PIII-500 is :  " & Format(MeanTimeEven / (j / 2), "0.000") & "<br>")
        Else
            MeanTimeOdd = MeanTimeOdd + (Timer - StartTime)
            Send ("Mean Time for PII-450 is :  " & Format(MeanTimeOdd / ((j + 1) / 2), "0.000") & "<br>")
        End If
        If i > TestQuantity - 2 Then
            totalbuf = totalbuf & buf
        End If
    Next i
    Send ("<hr>")
    Send (totalbuf)
    'Send (buf)
End If
End Sub

modWinInet.bas

Option Explicit

' Initializes an application's use of the Win32 Internet functions
Public Declare Function InternetOpen Lib "wininet.dll" Alias "InternetOpenA" _
(ByVal sAgent As String, ByVal lAccessType As Long, ByVal sProxyName As String, ByVal sProxyBypass As String, ByVal lFlags As Long) As Long

' User agent constant.
Public Const scUserAgent = "http sample"

' Use registry access settings.
Public Const INTERNET_OPEN_TYPE_PRECONFIG = 0
Public Const INTERNET_FLAG_ASYNC = &H10000000 ' this request is asynchronous (where supported)

' Opens a HTTP session for a given site.
Public Declare Function InternetConnect Lib "wininet.dll" Alias "InternetConnectA" (ByVal hInternetSession As Long, ByVal sServerName As String, ByVal nServerPort As Integer, ByVal sUsername As String, ByVal sPassword As String, ByVal IService As Long, ByVal lFlags As Long, ByVal lContext As Long) As Long

' Number of the TCP/IP port on the server to connect to.
Public Const INTERNET_DEFAULT_FTP_PORT = 21
Public Const INTERNET_DEFAULT_GOPHER_PORT = 70
Public Const INTERNET_DEFAULT_HTTP_PORT = 80
Public Const INTERNET_DEFAULT_HTTPS_PORT = 443
Public Const INTERNET_DEFAULT_SOCKS_PORT = 1080

' Type of service to access.
Public Const INTERNET_SERVICE_FTP = 1
Public Const INTERNET_SERVICE_GOPHER = 2
Public Const INTERNET_SERVICE_HTTP = 3

' Opens an HTTP request handle.
Public Declare Function HttpOpenRequest Lib "wininet.dll" Alias "HttpOpenRequestA" (ByVal hHttpSession As Long, ByVal sVerb As String, ByVal sObjectName As String, ByVal sVersion As String, ByVal sReferer As String, ByVal sAcceptTypes As String, ByVal lFlags As Long, ByVal lContext As Long) As Long

' Brings the data across the wire even if it locally cached.
Public Const INTERNET_FLAG_RELOAD = &H80000000

' Sends the specified request to the HTTP server.
Public Declare Function HttpSendRequest Lib "wininet.dll" Alias "HttpSendRequestA" (ByVal hHttpRequest As Long, ByVal sHeaders As String, ByVal lHeadersLength As Long, ByVal sOptional As Any, ByVal lOptionalLength As Long) As Integer

' Queries for information about an HTTP request.
Public Declare Function HttpQueryInfo Lib "wininet.dll" Alias "HttpQueryInfoA" (ByVal hHttpRequest As Long, ByVal lInfoLevel As Long, ByRef sBuffer As Any, ByVal lBufferLength As Long, ByRef lIndex As Long) As Integer

' The possible values for the lInfoLevel parameter include:
Public Const HTTP_QUERY_CONTENT_TYPE = 1
Public Const HTTP_QUERY_CONTENT_LENGTH = 5

225
Public Const HTTP_QUERY_EXPIRES = 10
Public Const HTTP_QUERY_LAST_MODIFIED = 11
Public Const HTTP_QUERY_PRAGMA = 17
Public Const HTTP_QUERY_VERSION = 18
Public Const HTTP_QUERY_STATUS_CODE = 19
Public Const HTTP_QUERY_STATUS_TEXT = 20
Public Const HTTP_QUERY_RAW_HEADERS = 21
Public Const HTTP_QUERY_RAW_HEADERS_CRLF = 22
Public Const HTTP_QUERY_FORWARDED = 30
Public Const HTTP_QUERY_SERVER = 37
Public Const HTTP_QUERY_USER_AGENT = 39
Public Const HTTP_QUERY_SET_COOKIE = 43
Public Const HTTP_QUERY_REQUEST_METHOD = 45

' Add this flag to the about flags to get request header.
Public Const HTTP_QUERY_FLAG_REQUEST_HEADERS = &H80000000

' Reads data from a handle opened by the HttpOpenRequest function.
Public Declare Function InternetReadFile Lib "wininet.dll" _
(ByVal hFile As Long, ByVal sBuffer As String, ByVal _
 iNumBytesToRead As Long, _
 iNumberOfBytesRead As Long) As Integer

' Closes a single Internet handle or a subtree of Internet handles.
Public Declare Function InternetCloseHandle Lib "wininet.dll" _
(ByVal hinet As Long) As Integer

' Queries an Internet option on the specified handle
Public Declare Function InternetQueryOption Lib "wininet.dll" Alias _
"InternetQueryOptionA" _
(ByVal hinternet As Long, ByVal lOption As Long, ByRef _
 sBuffer As Any, ByRef iBufferLength As Long) As Integer

' Returns the version number of Wininet.dll.
Public Const INTERNET_OPTION_VERSION = 40

' Contains the version number of the DLL that contains the Windows Internet
' functions (Wininet.dll). This structure is used when passing the
' INTERNET_OPTION_VERSION flag to the InternetQueryOption function.
Public Type tWinInetDLLVersion
  lMajorVersion As Long
  lMinorVersion As Long
End Type

' Adds one or more HTTP request headers to the HTTP request handle.
Public Declare Function HttpAddRequestHeaders Lib "wininet.dll" Alias _
"HttpAddRequestHeadersA" _
(ByVal hHttpRequest As Long, ByVal sHeaders As String, ByVal _
 lHeadersLength As Long, _
 ByVal lModifiers As Long) As Integer

' Flags to modify the semantics of this function. Can be a
combination of these values:

' Adds the header only if it does not already exist; otherwise, an
error is returned.
Public Const HTTP_ADDREQ_FLAG_ADD_IF_NEW = &H10000000

' Adds the header if it does not exist. Used with REPLACE.
Public Const HTTP_ADDREQ_FLAG_ADD = &H20000000
Replaces or removes a header. If the header value is empty and the header is found, it is removed. If not empty, the header value is replaced.

Public Const HTTP_ADDREQ_FLAG_REPLACE = &H8000000

'dwResult - the HINTERNET, DWORD or BOOL return code from an async API
dwError - the error code if the API failed
Public Type INTERNET_ASYNC_RESULT
    dwResult As Long
    dwError As Long
End Type

Public Declare Function InternetSetStatusCallback Lib "wininet.dll" _
(ByVal hinternet As Long, ByVal InternetCallback As Long) As Long

'status manifests for Internet status callback

Public Const INTERNET_STATUS_RESOLVING_NAME = 10
Public Const INTERNET_STATUS_NAME_RESOLVED = 11
Public Const INTERNET_STATUS_CONNECTING_TO_SERVER = 20
Public Const INTERNET_STATUS_CONNECTED_TO_SERVER = 21
Public Const INTERNET_STATUS_SENDING_REQUEST = 30
Public Const INTERNET_STATUS_REQUEST_SENT = 31
Public Const INTERNET_STATUS_RECEIVING_RESPONSE = 40
Public Const INTERNET_STATUS_RESPONSE_RECEIVED = 41
Public Const INTERNET_STATUS_CTL_RESPONSE_RECEIVED = 42
Public Const INTERNET_STATUS_PREFETCH = 43
Public Const INTERNET_STATUS_CLOSING_CONNECTION = 50
Public Const INTERNET_STATUS_CONNECTION_CLOSED = 51
Public Const INTERNET_STATUS_HANDLE_CREATED = 60
Public Const INTERNET_STATUS_HANDLE_CLOSING = 70
Public Const INTERNET_STATUS_REQUEST_COMPLETE = 100
Public Const INTERNET_STATUS_REDIRECT = 110
Public Const INTERNET_STATUS_INTERMEDIATE_RESPONSE = 120
Public Const INTERNET_STATUS_STATE_CHANGE = 200

Public Declare Function InternetQueryDataAvailable Lib "wininet.dll" _
(ByVal hinternet As Long, ByVal iNumberOfBytesAvailable As Long, _
ByVal iFlags As Long, ByVal IContext As Long) As Long

Private hInternetSession As Long
Private hInternetConnect As Long
Private hHttpOpenRequest As Long

Public Function HTTP_Client_Get( _
    ServerName As String, _
    ObjectName As String, _
    GetHeader As String _
) As String

    Dim bDoLoop As Boolean
    Dim sReadBuffer As String * 2048
    Dim iNumberOfBytesRead As Long
    Dim iRetVal As Boolean

hInternetSession = InternetOpen(scUserAgent, 
INTERNET_OPEN_TYPE_PRECONFIG, vbNullString, vbNullString, 0)
If Not CBool(hInternetSession) Then
    MsgBox "InternetOpen failed."
End If

hInternetConnect = InternetConnect(hInternetSession, ServerName, 
INTERNET_DEFAULT_HTTP_PORT, _ 
vbNullString, vbNullString, 
INTERNET_SERVICE_HTTP, 0, 0)
If hInternetConnect > 0 Then
    hHttpOpenRequest = HttpOpenRequest(hInternetConnect, "GET", 
ObjectName, GetHeader, vbNullString, 0, _ 
INTERNET_FLAG_RELOAD, 0)
    If CBool(hHttpOpenRequest) Then
        iRetVal = HttpSendRequest(hHttpOpenRequest, vbNullString, 0, vbNullString, 0)
        If iRetVal = True Then
            bDoLoop = True
            While bDoLoop
                sReadBuffer = vbNullString
                bDoLoop = InternetReadFile(hHttpOpenRequest, 
sReadBuffer, Len(sReadBuffer), lNumberOfBytesRead)
                HTTP_Client_Get = HTTP_Client_Get & Left$(sReadBuffer, lNumberOfBytesRead)
                If Not CBool(lNumberOfBytesRead) Then bDoLoop = False
            Wend
        Else
            MsgBox "HttpSendRequest call failed; Error code: " & 
Err.LastDllError & "."
        End If
    Else
        MsgBox "HttpOpenRequest call failed; Error code: " & 
Err.LastDllError & "."
    End If
Else
    MsgBox "InternetConnect call failed; Error code: " & 
Err.LastDllError & "."
End If

iRetVal = InternetCloseHandle(hInternetSession)

End Function

Public Function HTTP_Client_Post(  
    ServerName As String, _
    ObjectName As String, _
    PostData As String, _
    PostDataLen As Long, _
    PostHeader As String, _
    PostHeaderLen As Long _
) As String

Dim bDoLoop As Boolean
Dim sReadBuffer As String * 2048
Dim lNumberOfBytesRead As Long
Dim iRetVal As Boolean

hInternetSession = InternetOpen(scUserAgent, 
INTERNET_OPEN_TYPE_PRECONFIG, vbNullString, vbNullString, 0)
If Not CBool(hInternetSession) Then
    MsgBox "InternetOpen failed."
End If

hInternetConnect = InternetConnect(hInternetSession, ServerName,
    INTERNET_DEFAULT_HTTP_PORT, vbNullString, vbNullString,
    INTERNET_SERVICE_HTTP, 0, 0)
If hInternetConnect > 0 Then
    hHttpOpenRequest = HttpOpenRequest(hInternetConnect, "POST",
        ObjectName, vbNullString, vbNullString, vbNullString, _
        INTERNET_FLAG_RELOAD, 0)

    If CBool(hHttpOpenRequest) Then
        iRetVal = HttpAddRequestHeaders(hHttpOpenRequest,
            PostHeader, PostHeaderLen, HTTP_ADDREQ_FLAG_REPLACE Or
            HTTP_ADDREQ_FLAG_ADD)
        iRetVal = HttpSendRequest(hHttpOpenRequest, PostHeader,
            PostHeaderLen, PostData, PostDataLen)
        If iRetVal = True Then
            bDoLoop = True
            While bDoLoop
                sReadBuffer = vbNullString
                bDoLoop = InternetReadFile(hHttpOpenRequest,
                    sReadBuffer, Len(sReadBuffer), lNumberOfBytesRead)
                HTTP_Client_Post = HTTP_Client_Post &
                    Left$(sReadBuffer, lNumberOfBytesRead)
                If Not CBool(lNumberOfBytesRead) Then bDoLoop =
                    False
            Wend
            Else
                MsgBox "HttpSendRequest call failed; Error code: " & Err.LastDllError & ","
            End If
        Else
            MsgBox "HttpOpenRequest call failed; Error code: " & Err.LastDllError & ","
        End If
    Else
        MsgBox "InternetConnect call failed; Error code: " & Err.LastDllError & ","
    End If

iRetVal = InternetCloseHandle(hInternetSession)
End Function