

A wireless solution to the manual processes  
used by breakdown companies

Randeep Chahal  
Computer Science  
(2002/2003)

The candidate confirms that the work submitted is their own and the appropriate credit has been given where reference has been made to the work of others.

I understand that failure to attribute material which is obtained from another source may be considered as plagiarism.

(Signature of student)

---

---

## Summary

---

In recent times a new paradigm of computing has been fashioned called mobile computing. Advances in portable information appliances and hardware technologies, such as mobile phones, personal digital assistants (PDA), handheld and portable computers and wireless networks have lead to this creation. This project develops a mobile application for vehicle recovery companies in an attempt to understand and evaluate the mobile computing arena.

The application caters for data collection and storage in the work field environment. It can easily be adapted for any business that has a mobile workforce and was developed on the Palm OS.

## Acknowledgements

---

I would like to thank the following people for their support, advice and help during work on this project: My supervisor Haiko Muller, my housemates Dave, Keith and Kit and my Parents.

I would especially like to say thanks to Steve Willingham for providing the information and agreeing to be a potential user.

<b>1. INTRODUCTION.....</b>	<b>6</b>
1.1 PROJECT AIM .....	6
1.2 REASONS FOR PROJECT CHOICE.....	6
1.3 OBJECTIVES .....	6
1.4 MINIMUM REQUIREMENTS .....	6
1.5 POSSIBLE ENHANCEMENTS.....	7
<b>2. BACKGROUND .....</b>	<b>8</b>
2.1 INTRODUCTION TO MOBILE COMPUTING.....	8
2.2 INTRODUCTION TO THE VEHICLE RECOVERY INDUSTRY .....	9
2.3 INTRODUCTION TO THE AUTOMOBILE ASSOCIATION (AA) .....	10
2.4 INTRODUCTION TO THE ROYAL AUTOMOBILE CLUB (RAC).....	10
<b>3. PROJECT MANAGEMENT.....</b>	<b>11</b>
3.1 METHODOLOGIES .....	11
3.1.1 <i>Rapid Application Development (RAD)</i> .....	11
3.1.2 <i>The Waterfall Model</i> .....	11
3.1.3 <i>The V-Process Model</i> .....	11
3.1.4 <i>The Spiral Model</i> .....	11
3.2 CHOSEN METHODOLOGY .....	11
3.3 MILESTONES .....	12
3.4 PROJECT SCHEDULE .....	12
3.5 STRUCTURE OF REPORT.....	13
<b>4. ANALYSIS OF SYSTEMS.....</b>	<b>15</b>
4.1 FEASIBILITY STUDY .....	15
4.1.1 <i>Problem Definition</i> .....	15
4.1.2 <i>Economic Feasibility</i> .....	15
4.1.3 <i>Technical Feasibility</i> .....	16
4.1.4 <i>Operational Feasibility</i> .....	16
4.2 RESEARCH METHODS.....	16
4.3 REQUIREMENTS ANALYSIS.....	17
4.3.1 <i>The AA's Existing System</i> .....	17
4.3.1.1 <i>The Manual System</i> .....	17
4.3.1.2 <i>Weaknesses of the Manual System</i> .....	18
4.3.1.3 <i>Attempts to Improve Overall System</i> .....	18
4.3.1.4 <i>Conclusions</i> .....	18
4.3.2 <i>The RAC's Existing System</i> .....	19
4.3.2.1 <i>The System</i> .....	19
4.3.2.2 <i>Weaknesses of the System</i> .....	19
4.3.2.3 <i>Conclusions</i> .....	19
4.3.3 <i>Examination of Other Work Field Systems</i> .....	20
4.3.3.1 <i>E-Trace</i> .....	20
4.3.3.2 <i>OmniExpress</i> .....	20
4.3.3.3 <i>Other Systems</i> .....	21
4.3.3.4 <i>Conclusions</i> .....	21

<b>5. REQUIREMENTS SPECIFICATION .....</b>	<b>22</b>
5.1 THE PROBLEM OWNER .....	22
5.2 THE INTENDED USERS .....	22
5.3 REQUIRED SYSTEM PROCESSING.....	22
5.3.1 <i>Essential System Processing</i> .....	22
5.3.2 <i>Desirable System Processing</i> .....	22
5.3.3 <i>Further System Processing</i> .....	23
5.4 DATA MODELS .....	23
5.4.1 <i>Data Flow Diagram</i> .....	23
5.4.2 <i>Logical Data Structure</i> .....	23
5.5 FUNCTIONAL REQUIREMENTS .....	23
5.5.1 <i>Essential Functional Requirements</i> .....	23
5.5.2 <i>Desirable Functional Requirements</i> .....	24
5.5.3 <i>Further Functional Requirements</i> .....	24
5.6 NON-FUNCTIONAL REQUIREMENTS .....	24
5.6.1 <i>Essential Non-Functional Requirements</i> .....	24
5.6.2 <i>Desirable Non-Functional Requirements</i> .....	24
5.6.3 <i>Further Non-Functional Requirements</i> .....	25
<b>6. DESIGN .....</b>	<b>26</b>
6.1 TECHNICAL SYSTEM OPTIONS.....	26
6.1.1 <i>Identification of Constraints</i> .....	26
6.1.2 <i>Front End Hardware Options</i> .....	26
6.1.2.1 <i>Operating Systems</i> .....	27
6.1.2.2 <i>Wireless transfer methods</i> .....	29
6.1.3 <i>Chosen Front End Hardware</i> .....	29
6.1.4 <i>Development Language and Environment Options</i> .....	30
6.2 DATA DESIGN .....	31
6.2.1 <i>Relational Data Analysis</i> .....	31
6.2.2 <i>Data Flow Diagram</i> .....	31
6.2.3 <i>Data Structure</i> .....	31
6.3 PROCESS DESIGN .....	32
6.4 USER INTERFACE DESIGN .....	33
<b>7. IMPLEMENTATION .....</b>	<b>35</b>
7.1 DEVELOPMENT ENVIRONMENT .....	35
7.1.1 <i>GNU PRC-Tools</i> .....	35
7.1.2 <i>Cygwin</i> .....	35
7.1.3 <i>Palm OS SDK v.5</i> .....	36
7.1.4 <i>Palm OS Emulator</i> .....	36
7.1.5 <i>Palm IIIe</i> .....	36
7.2 OUTLINE OF DEVELOPMENT .....	36
7.2.1 <i>Resources used</i> .....	36
7.2.2 <i>Palm File Structure</i> .....	36
7.2.3 <i>Palm Application Structure</i> .....	37
7.2.4 <i>Storage of Data on Palm OS</i> .....	38

7.2.5 Data Compression .....	38
7.2.6 Drawing Functions .....	39
7.2.7 Categories .....	39
7.3 RECOVERY SCREENS AND MENUS .....	40
7.3.1 Recovery List .....	40
7.3.2 Member Edit.....	42
7.3.3 Breakdown Details .....	43
7.3.4 Note Form.....	44
7.3.5 Menus.....	44
7.4 PROBLEMS AND ISSUES .....	45
<b>8. EVALUATION .....</b>	<b>47</b>
8.1 TESTING.....	47
8.1.1 White box testing.....	47
8.1.2 Black Box Testing.....	48
8.1.3 User Acceptance Testing.....	48
8.2 FUTURE IMPROVEMENTS .....	49
8.3 EVALUATION OF CHOSEN METHODOLOGY .....	50
8.4 EVALUATION OF FRONT END HARDWARE.....	50
8.5 EVALUATION OF DEVELOPMENT LANGUAGE .....	51
8.6 EVALUATION OF DEVELOPMENT ENVIRONMENT.....	52
8.7 EVALUATION OF S YSTEM .....	52
8.9 EVALUATION OF THE FUTURE OF MOBILE COMPUTING.....	54
8.10 CONCLUSIONS .....	55
<b>REFERENCES.....</b>	<b>56</b>
<b>APPENDIX A: THE REFLECTION .....</b>	<b>59</b>
<b>APPENDIX B: MILESTON ES .....</b>	<b>621</b>
<b>APPENDIX C: PROJECT SCHEDULE .....</b>	<b>62</b>
<b>APPENDIX D: INTERVIEW WITH AA PATROL MAN, STEVE WILLINGHAM .....</b>	<b>64</b>
<b>APPENDIX E: FORMS USED BY AA PATROLS AND THE SYSTEMS THESE ARE ENTERED INTO .....</b>	<b>67</b>
<b>APPENDIX F: LOGICAL DATA STRUCTURES, DATA FLOW DIAGRAMS.....</b>	<b>69</b>
<b>APPENDIX G: DEVICE COMPARISON .....</b>	<b>73</b>
<b>APPENDIX H: ENTITY EVENT MATRIX, ENTITY LIFE HISTORY DIAGRAM, DATA CHANGES .....</b>	<b>76</b>
<b>APPENDIX I: PALM APPLICATION EVENT LOOP .....</b>	<b>82</b>
<b>APPENDIX J: GREMLINS TEST RESULTS .....</b>	<b>83</b>
<b>APPENDIX K: USER ACCEPTANCE TESTING: 2<sup>ND</sup> INTERVIEW WITH AA PATROL MAN, STEVE WILLINGHAM.....</b>	<b>88</b>
<b>APPENDIX L: RECOVERY: USER MANUAL .....</b>	<b>91</b>

# 1. Introduction

---

## 1.1 Project Aim

The project will explore the possible solutions to data capture and transfer in work field environments, specifically looking at the data captured by vehicle breakdown companies on the road side. The project will investigate which mobile front ends are currently available and which of these are most suitable for recovery patrols. It will also look at the most appropriate methods of data storage and transfer from mobile devices. The end aim is to produce an information system, which replaces and improves the current manual processes used by breakdown companies and use this to assess the future of mobile computing.

## 1.2 Reasons for Project Choice

"Intelligent wireless handheld devices are going to explode, absolutely explode over the next several years."

Steve Ballmer, CEO, Microsoft

The project was chosen, as mobile computing is a field that is relatively young and is currently growing (Refer to section 2.1). Many believe, as Steve Ballmer, that field of mobile computing is the next evolutionary step in the development of computers. It is perceived as radically changing, as revolutionary as traditional networks were when they were first introduced. It was therefore an ideal topic for a project, as the project will assess the future of the field through the development of a mobile solution.

## 1.3 Objectives

The objectives of the project are:

- To research the current manual processes used on the road by patrols and explore the problems they face.
- To research and explore the possible hardware and software solutions to data capture on mobile computers.
- To implement a front end solution to replace the manual processes.
- To evaluate the solution and review how it improves the current processes.
- To review how the solution could be extended.
- To reflect on the future of mobile computer technology and what possible applications wireless devices could be used for.

## 1.4 Minimum Requirements

The minimum requirements of the project are:

- The front end should be portable and have the ability to capture and store information such as basic vehicle details, basic member details and simple repair notes that are input by patrol vehicles when they respond to a call.
- This information should then be transferred from the hand-held device to a desktop PC via a physical cable, for example a docking station.

### **1.5 Possible Enhancements**

The possible enhancements to the project are:

- Store the information in a database, which holds details of all the breakdown jobs and members. This database is located on the desktop PC.
- Allow the ability to have all members held on the desktop and only those required for that day to be held on the handheld.
- Allow more technical information such as possibility of damage, vehicle condition etc to be captured.
- Allow customers to apply for membership via the front-end communication module.
- Allow details of parts used to be recorded for later invoicing.
- Storing the database on a server and transferring the information from the current desktop to the server via the Internet.



## 2. Background

---

### 2.1 Introduction to Mobile Computing

In recent times a new paradigm of computing has been fashioned called mobile computing. According to [1] [2] advances in portable information appliances and hardware technologies, such as mobile phones, personal digital assistants (PDA), handheld and portable computers and wireless networks have lead to this creation. Mobile computing is a large field that spans various areas of computer science and engineering. These include wireless networking, distributed systems, operating systems, distributed databases, software engineering and applications development to name a just a few. The project will touch on many of these areas although it will not be exploring each of them in great detail, only those which are relevant.

Mobile computing is currently growing at a fast rate, [3] believes that by 2006 the number of data-enabled phone, PDA and other device users is expected exceed the worldwide Internet sub-scriber population. Especially in business mobile devices are seen as essential tools for increasing productivity in mobile workers and keeping one step ahead of the competition:

"By 2004/5, we expect 65%-75% of enterprise to deploy extension to mission critical applications for wireless and/or pervasive platforms, and expect 75% of corporate knowledge workers to be mobile at least 25% of the time."

Meta Group

One factor leading to this adoption is the price of easy to handle devices. It is rapidly falling while the benefits of such devices are increasingly apparent in both consumer and business settings. More and more, the business user is looking to use mobile devices to perform tasks that previously could only be handled by the desktop PC. At the same time hardware and software manufacturers are adapting their products for mobile use fuelling the current charge for all things wireless.

Developing software and hardware for these new mobile computing systems creates new challenges that are hardly trivial. These challenges are quite different from those involved in the design of today's stationary-networked systems [4]. Factors that need to be taken into account include wireless communication, battery life, size and storage restrictions, user interface constraints and security issues. These will be described further in chapter 6.

## 2.2 Introduction to the Vehicle Recovery Industry

This type of project could have been implemented in many industries that require workers to operate in work field environments. Any industry that requires their employees to collect and store information outside of the main office would suffice. Figure 1 shows some of the industries that mobile computing could benefit.

Many of these industries have already adopted mobile computing and for some it has even transformed the way they operate. The examples of FedEx and UPS (companies in the delivery industry) are listed in [3]. They have been using mobile computing for some time, by adopting mobile terminals for their employee's they have revolutionised the delivery business.

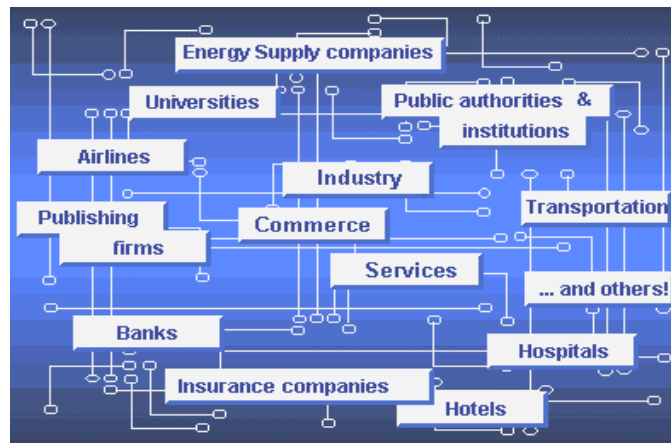


Figure 1: Industries that could use mobile computing

[Source: Zaslavsky A (2001): Mobile Computing at Monash University]

One such industry that is yet to use the full benefit of mobile computing is the vehicle recovery industry. Vehicle recovery is the business of recovering and where possible repairing vehicles when they break down. These companies often have membership schemes where only members are capable of receiving a 'call out'. Many of the companies in the industry already use GPS tracking and mobile data terminals to direct and keep track of the patrols. However simple data collection, storage and transfer in some cases still manual. One problem is that many of the current mobile data terminals are bespoke and will not cater for such collection.

There are numerous vehicle breakdown services in the UK however the largest ones are the AA [5], RAC [6] and Green Flag [7]. Therefore the project will particularly concentrate on the AA and the RAC as more information on their business processes is available, see section 4.2 for further details. Currently all of these still have some manual processes, which range from membership details to parts and service lists.

### **2.3 Introduction to the Automobile Association (AA)**

The AA is the UK's largest motoring organisation with millions of members and some 3,600 highly trained patrols. Originally formed by a group of motoring enthusiasts in 1905 intending to help motorists avoid police speed traps, it now offers a number of services from recovery to personal loans. The project will however only be concerned with the recovery aspect of the business.

### **2.4 Introduction to the Royal Automobile Club (RAC)**

Founded in 1897, the RAC now has six million members, who can access a large range of motoring products and services. These include roadside assistance in the event of a breakdown to continually updated legal and technical advice and up-to-the-minute travel information. They also claim to have "the world's most advanced computer systems to deal with calls for roadside assistance". Once again the project will only be concerned with the recovery aspect of the business.

## 3. Project Management

---

### 3.1 Methodologies

The choice of methodology plays an important role in the success of a project, especially in project that requires the development of software where requirements may change. It therefore makes sense to outline a suitable methodology before the design stage of the project. The choice of methodology can be divided into two stages; the first stage is to select a process model and the second stage is to select a methodology that uses or can be adapted to that process model. The following summary on process models is based on ([8], Chapter 4) and the methodologies were taken from ([9], Chapter 6):

#### 3.1.1 Rapid Application Development (RAD)

This process is designed to reduce the development time by removing some of the processes seen as bureaucratic. The idea of the process is to quickly develop a system, which has limited functionality but is accepted by the user. This can then be enhanced to its complete functionality, without the need to go through phases of user feedback and redesign. One disadvantage of RAD is that it does not fully support the analysis and requirements processes.

#### 3.1.2 The Waterfall Model

The waterfall model is a simple eight-stage model, where each stage follows on from the one before. The simplicity of the model is one of its main advantages however it has limited scope for iteration and therefore is more suitable for projects where there is a definite specification that is unlikely to change.

#### 3.1.3 The V-Process Model

The V-process model is very much like the waterfall model; however it expands the testing activities and stresses the necessity for validation. The biggest advantage of this model is that it insists on going through phases of user feedback and then corrections. This produces a system that will be accepted by the user, but requires large amounts of user communication and is very orientated towards one set of users.

#### 3.1.4 The Spiral Model

The spiral model is another way of looking at the basic waterfall model, however there is a larger level of detail considered at each stage. The system is considered in more and more detail in each sweep and an evaluation process is undertaken before the next iteration is embarked upon. One advantage of this model is that it can easily be adapted and modified to suit the specific needs of a project.

### 3.2 Chosen Methodology

For this project, an iterative process is more suitable as the project scope may change and therefore the requirements may also change. By using a modified spiral model combined with a methodology, each stage of the project can be

---

reviewed and possible changes can be implemented. In terms of user interface building however RAD will provide the ability to build the user interface without having to communicate with the users at each stage, which is a known difficulty in this project. This will ensure that the user interface design does not impact the progress of the projects investigation.

The modified spiral model will be based on the Structured Systems Analysis and Design Methodology (SSADM) version 4 (see figure 2). As the system to be produced is concerned with the collection of data, a data orientated methodology seemed to be more appropriate. SSADM is data orientated and can easily be adapted to the spiral model. In addition not all parts of the methodology are compulsory which means that it can be scaled down. This is important, as a project of this size does not require such in depth processes. It is also a 'tried and tested' methodology, all government departments' use it as was developed by the Central Computing and Telecommunications Agency (CCTA).

Other methodologies investigated included Merise, Object Orientated Analysis (OOA), Rational Unified Process (RUP), Euromethod (EM) and Jackson Systems Development (JSD). The object orientated methodologies OOA and RUP were deemed unnecessary as the system would not be object orientated. The other methodologies were not data orientated or easily adapted to the spiral model.

As mentioned above a cut down version of SSADM will be used and diagrammatic representations of the system will not be used at all stages. Many of the stages will be altered to reflect the management issues of a student project. This includes removing some stages such as Define Business Options. As there is no specific business, Detailed Cost Benefit Analysis and the creation of multiple Business Options is not required and is beyond the scope of a student project.

### **3.3 Milestones**

The milestones of the project can be seen in Appendix B. These milestones were later revised when the scope of the project changed to focus more on mobile computing research and producing software for the front-end device. Appendix B contains the revised milestones.

### **3.4 Project Schedule**

The original project schedule can be found in Appendix C. This project schedule was later revised when the scope of the project changed to focus more on mobile computing research and producing software for the front-end device. Appendix C contains the revised project schedule.

**Note:** As the scheduled date for completing the standard functionality (only satisfying the minimum requirements) came closer it became apparent that there would not be enough time to implement any extra functionality. This was because the standard functionality proved far more complex than originally thought, see chapter 7 for details. The time allocated for the extra functionality was therefore allocated to the standard functionality.

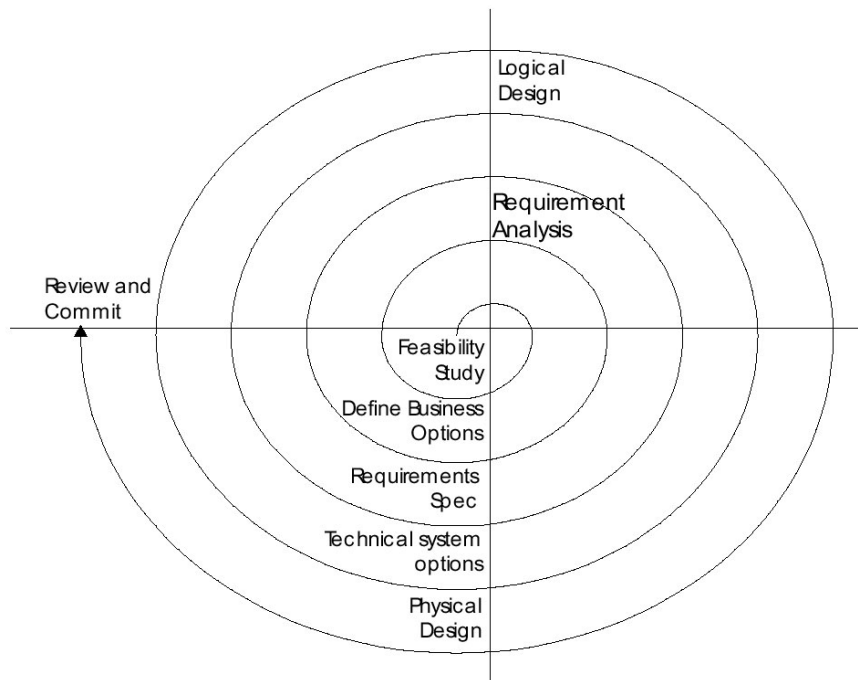


Figure 2: The application of the spiral model to SSADM version 4.

### 3.5 Structure of Report

The remainder of this report will be split up into the following chapters:

#### Chapter 4: ANALYSIS OF SYSTEM

Contains the Feasibility Study stage of the SSADM model, which outlines some of the projects limitations. Section 4.3 correlates to the Requirements Analysis stage, which details the existing system and other similar systems.

#### Chapter 5: REQUIREMENTS SPECIFICATION

Lists the hardware and software requirements produced from the analysis of the existing system and research into other systems. This chapter correlates to the Requirements Specification stage of SSADM.

#### Chapter 6: DESIGN

Section 6.1 details the possible development options and the eventual selections; this correlates to the Technical System Options stage of SSADM. Sections 6.2 onwards detail the various design stages that include data, process and user interface design. These correlate to the Logical and Physical Design stages of SSADM.

#### Chapter 7: IMPLEMENTATION

Discusses the development of the application and problems encountered during the implementation.

#### Chapter 8: EVALUATION

Deals with the testing and evaluation of the application and setting out a scope for future work. Draws a conclusion on the whole project, reviewing the projects achievements against initial objectives.

In addition, the appendices contain additional information including the applications user manual and interview write-ups.

## 4. Analysis of Systems

---

### 4.1 Feasibility Study

The following section is a simple analysis of the problem, which was used to decide whether to proceed to a full study. It also outlines whether to proceed in a different direction from that envisaged in the initial problem and what limitations exist on resolving the problem.

#### 4.1.1 Problem Definition

There are many businesses, which have employees who have to collect data in the work field environment whether that is in a client's office or out on the roadside. All these businesses have some problems in common, how to collect this data, how to transfer it, where to store it and how to use it effectively to improve their businesses. With the new portable devices out on the market and the new emerging wireless technologies these problems can at last be solved at a cost that is acceptable.

One such sector of business that has these problems is the vehicle recovery sector, information about the breakdowns has to be recorded, transferred and stored. Currently these processes are manual and could be computerised. This project will therefore attempt to solve the general problems stated in the above paragraph by designing and building a system for use by the vehicle recovery sector. By doing this it will investigate the new technologies mentioned above and evaluate their possible uses and future.

#### 4.1.2 Economic Feasibility

As the project has not been initiated by a company there is no budget and hence the software tools that are to be used have to be either freely available or university owned. There also cannot be a hardware cost, therefore any testing and development must be done on existing university machines, on available emulators, or on a device that is freely available. As there is a restricted time period, a large system to resolve all the problems mentioned in section 4.1.1 cannot be produced. Instead a 'cut down' version can be developed that solves some of the problems. This also means there is a limited time for learning and hence development languages and techniques chosen should be those already known where possible.

Theoretically if this was to be implemented by an actual company the benefits should out way the costs. Therefore theoretical hardware costs should be kept to a minimum. If the system was actually implemented the expected benefits would be:

- Improved customer service, through the storage of previous breakdown details.
- Removal of physical storage.
- Better knowledge of customers, as their details are stored and can be analysed.



- Elimination of repeated data entry.
- Patrols have an extra level of insurance as they now have a permanent record of the breakdown work carried out.

#### *4.1.3 Technical Feasibility*

As the main objective of this project is to produce a completely new system we only need to consider possible methods of outputting the data for use by another system. We do not have to actually integrate the system with any particular existing system. If information needs to be output to another system then a universal method such as XML should be used. Therefore there are no restraints on the technology that can be used apart from that it should be freely available for use.

The system should also be feasible in the set time period; it cannot therefore be too technical. This means that certain complex processes of examined vehicle recovery companies will have to be left out initially. They can however be added later if there is time remaining.

#### *4.1.4 Operational Feasibility*

The system to be produced will initially have reduced functionality than that required by an actual breakdown recovery company. More functionality such as additional screens will be added later, however an actual breakdown company would require more advanced features. For example they may want continuous wireless connection, integration with their GPS satellite navigation and even the possibility of engine analysis. These are far beyond the scope of this project; the system to be produced would only be a starting place for such integrated systems.

## **4.2 Research Methods**

Extensive research was carried out to gather information on the recovery companies systems and processes and to examine other work field systems. The following research methods were considered:

- Questionnaires/Surveys
- Interviews
- Documentation Review
- Observation
- Case Studies

The first task was to find a suitable contact in a recovery company who could provide the information that was required. This proved very difficult with the larger organisations as many would only have one method of contacting them and this would be via their customer service department. After numerous phone calls only one of the companies the RAC, provided a name and address of a contact who could supply details of their systems.

To ensure that the project had a potential user Steve Willingham, an AA patrolman was contacted. Mr Willingham is the father of a friend and agreed to help provide information on the AA's systems. A letter was sent to Ken Hartley the Information Science Development Manager at RAC and after some time a response was received. Mr Hartley's letter detailed the RAC's existing system, which was already highly computerised. Due to this fact, and the lack of other available resources it was decided that the project would use the AA's existing system as a basis for the new systems design and use Mr Willingham as a potential user.

As there was only one source of information it was decided that questionnaires were unnecessary and that interviewing Mr Willingham would be more suitable so that a full range and depth of questions could be asked. Observational techniques could not be used as Mr Willingham lives in Blackpool, however Mr Willingham provided documents that had to be completed at each breakdown, so document reviews were possible. To ensure the system developed was in keeping with the requirements of other companies who operate in work field environments, case study examples were examined. These can be seen in section 4.5.

### **4.3 Requirements Analysis**

This section will aim to provide an overview of the research into the current manual systems used by patrols and identify problems or areas that need improvement. It will aim to outline and explore the methods and tools that can be used to produce a solution to the problem by investigating similar systems.

#### *4.3.1 The AA's Existing System*

Steve Willingham provided all details about the AA's existing system during an Interview over the phone on the 14<sup>th</sup> of November 2002. A write up of the phone interview together with the questions asked can be seen in Appendix D.

##### **4.3.1.1 The Manual System**

The AA currently has a number of forms, which can be filled in by the patrols. These are the breakdown report, membership application form and an Invoice for parts/service. Photocopies of these can be seen in Appendix E.

The Breakdown Report has to be filled in on each breakdown attended. The top copy of the report is given to the customer while a second copy is given to either the garage the vehicle is taken to or the second patrol if one is required (see Appendix E). The third copy of the form is kept by the original patrol. If parts are used in repairing the vehicle then an Invoice for Parts/Services form has to be completed. If there is someone who requires a membership then the Membership Application Form is filled in. These also have multiple copies, which can be given to customers, garages, other patrols, suppliers, accounts and the membership department. See section 5.4.1, which explains a data flow diagram, which shows how the data flows through the AA's current system.

The data collected by the patrols is then entered into one of three separate systems. One system stores information on deployment, the second system stores information on breakdowns and the other system stores information on memberships.

#### 4.3.1.2 Weaknesses of the Manual System

According to AA Mr Willingham, the existing system creates a "large paper trail" which becomes difficult to manage and store. Presently the Breakdown Report is not even used effectively as there is no computer system in place to store this data hence the forms are kept by the patrols and stored for three to six months and then destroyed. Also the design of the breakdown forms has caused some patrolmen to complain; as they believe the form is too complex and time consuming to fill in at each breakdown.

The systems that store the data collected have the ability of communicating to one another however they are still separate systems, all information on a particular member and their breakdown history is not stored in one central system. This means that when a member rings up the call centre to lodge a breakdown, the call operative does not have any information on previous breakdowns only the member's details.

#### 4.3.1.3 Attempts to Improve Overall System

Currently the AA is in the process of redesigning their business processes and overhauling their entire operation by implementing supply chain management systems. They have piloted many schemes to improve their customer relationship management and improve their repair rate; this has included the recent (one-year) introduction of the Breakdown Report forms and introduction of engine analysis through laptops. The Breakdown Reports have been introduced so that the AA can have a record of the breakdown in case a member makes a complaint or tries to claim compensation for something they believe was caused by the patrol.

#### 4.3.1.4 Conclusions

Mr Willingham's ideal solution to the problems would be to produce an integrated system that caters for engine analysis, data capture, route guidance, and email facilities. This would preferably be implemented on the existing laptop and would have wireless connectivity to all the AA's backend systems so that data collected would be directly transferred without the need for patrols to physically take the forms. Steps to provide this type of system have already started as mentioned in the above section.

This type of solution is far beyond the scope of the project and it was agreed that a simpler system that caters for data collection and resolves the issues of the manual system would be a better direction for the project. The project will

attempt to replace the existing manual forms and provide a method of storing these forms. It will be a system that could not only be used as a basis for the AA but also by other vehicle recovery companies.

#### *4.3.2 The RAC's Existing System*

Ken Hartley, the Information Science Development Manager at the RAC, supplied details about the RAC's existing system.

##### 4.3.2.1 The System

The RAC still have some manual processes however they are far more advanced in terms of their computerisation of their business processes than the AA. In May 2001 they gave each of their 1400 patrols their own laptop PC called the Patrol PC, which has a variety of uses. They use the PC to look up information in order to be able to fix vehicles at the roadside. Stored on the PC is a large amount of technical data for car models and information on towing, health and safety, operations procedures, parts information, etc. The PC is also used for diagnostic procedures on vehicle electronics, very similar to the proposed laptop piloted by the AA.

The RAC do not collect data on the actual breakdown as the AA do, they simply store a fault code. When a breakdown is assigned details such as member information and vehicle make and model are sent to a Mobile Data Terminal. When the repair is complete the patrol uses the terminal to enter a fault code relating to the breakdown and make themselves available for the next job.

##### 4.3.2.2 Weaknesses of the System

Applying for membership is still done manually however the process is soon to be incorporated into the front-end application on the Patrol PC. The membership form once incorporated will have the ability to access the customer relationship database held at the Bristol office.

A laptop PC is heavy, cumbersome, and has poor battery life. It requires the patrols to have somewhere to place the laptop whether that be on their lap when they are sitting down or on some flat, stable surface.

##### 4.3.2.3 Conclusions

As the project is focusing on the removal of manual processes to improve the process flow the RAC will only be used as a source for further improvements. The RAC have already removed almost all manual processes and do not collect customer or breakdown information at each breakdown, therefore the project shall be focused more towards the using the AA system as a basis.

#### *4.3.3 Examination of Other Work Field Systems*

Currently there are numerous systems being developed and used in work field environments. These range from systems to store information on deliveries to more advanced systems that are used not only for data capture but also for communication and global positioning. Although many of these systems are beyond the scope of this project they provide an insightful look into the platforms used for development and the hardware choices available. In particular the solutions researched have been for businesses that have mobile workers.

##### *4.3.3.1 E-Trace*

One system called eTrace, developed by Gearworks [10], was the closest system to resolving the problem definition (see section 4.1.1). It is a generic product produced for “dispatched mobile workers”, however it is “modular and elastic” so that it can be customised to the particular client.

eTrace is a web based wireless application that runs on either Windows Pocket PC or Palm OS; this means that it operates on a handheld device. The front end of the system has SmartForms, which acts as an electronic clipboard, which can be used to enter any repetitive data such as customer information or vehicle details. This data can then be exchanged in real time with the central office. The back end of the system is a Mission Control for dispatchers to plan, route and measure the performance of the mobile workforce. The most interesting aspect of this software is that it is built on open standards, which means it can integrate with existing systems easily. It has a software module that allows integration to an enterprise system through XML.

##### *4.3.3.2 OmniExpress*

One of the more advanced systems researched was OmniExpress developed by Qualcomm, the world’s largest mobile data solutions provider [11]. This gave some insight into what can be produced with a large development team and substantial investment. OmniExpress is a system designed for large fleet operators who would like to communicate with and locate their vehicles at all times. It takes advantages of the latest technology available in the USA: CDMA to transmit data and GPS to relay positioning information. This does however mean the system requires specialist hardware. This includes a GPS antenna used to position the vehicle, a TCU used to transmit and receive information and an in-vehicle computer. As with many of the systems researched the in-vehicle computer ran on the Windows CE OS, which allowed for a degree of customisation and gives OmniExpress the ability to operate in conjunction with other systems on one unit.

#### 4.3.3.3 Other Systems

Other systems researched included Wireless Vehicle Inspection software used to capture information and images of vehicles, a Mobile Point of Sale system used on the forecourts of car dealerships and a system for technology engineers to input service information.

#### 4.3.3.4 Conclusions

After analysing the systems it is clear that there are many methods of approaching the problem and developing a system. There is yet no single method of implementing a system of this type as the technology is continuously moving and hence implementations and their capabilities vary. Many systems that have been developed are deployed on 'rugged' hardware such as those produced by Symbol technologies [12]. This is because operating in a work field environment can be hazardous and normal devices are not built to endure such environments. For example a device used at the side of a road has to be 'rugged' as it has to withstand being moved in and out of a vehicle, being handled by various people and being exposed to the 'elements' (wind, rain, dust etc).

It is clear from the systems investigated that software producers are creating solutions for the processes of both data collection and workforce management. Workforce management is a possible extension to the system that could be implemented if more time was available. Most of the systems run on either Windows operating systems or Palm operating systems. Older Windows based systems ran on Windows CE whilst newer systems ran on Windows Pocket PC. In comparison the systems that ran on the Palm operating system were more stable as the Palm OS does not vary between versions as Windows does. The hardware used to run these operating systems varies considerably from specialised hardware developed specifically for the software to 'off the shelf' Pocket PC's.

A general theme noticed in the various systems was that they all offered a degree of integration with existing systems. This was due to their development being based on Internet technologies and standards. The eTrace system mentioned earlier uses a software module called QuickConnect that converts the data output into XML so that it can be read by any existing system that has a standard XML connection. XML seems to be the industry standard for integration, one company @Road [13] has even produced a set of Application Program Interface's (API) so that any combination of its modules can be chosen and integrated to the existing system. In terms of the project if the data was transferred from the front-end device to a backend database system, this transfer could be done via XML. This would allow the front-end device to connect to any device that can understand the XML standard. This however would take some time and could not be done in the time provided. It could however be a possible enhancement for the future.

## 5. Requirements Specification

---

This section outlines the specific requirements for the solution. These are to be used in section 6.1, the Logical System Specification to select the most appropriate hardware and software etc to be used. They will also be utilised as criteria for the evaluation of the final solution.

### 5.1 The Problem Owner

Any company that requires their mobile workforce to collect and store information experiences the problem of data collection in the work field environment, see section 4.1.1 for more details. More specifically some breakdown recovery companies such as the AA have problems collecting data on their members and breakdowns, see section 4.3.1.2 for more details.

### 5.2 The Intended Users

The intended users are any recovery company that requires data collection by their employees in a work field environment. This can however be expanded to any business or even individual which needs to collect information, store it and then transfer it.

### 5.3 Required System Processing

As the system will be based on a mobile computer the processing tasks should be kept at a minimum, as mobile devices tend to be less powerful than desktops etc. Wherever possible, processing tasks should be completed on the machine the mobile device connects to rather than the mobile device itself. This machine could be a desktop or a server.

#### 5.3.1 Essential System Processing

The essential processing tasks are as follows:

- Creation of member details on the mobile device.
- Creation of vehicle (members) details on the mobile device.
- Updating of vehicle (members) details on the mobile device.
- Creation of a breakdown report on the mobile device.
- Updating of current breakdown report on the mobile device.

#### 5.3.2 Desirable System Processing

The desirable processing tasks are as follows:

- Creation of invoice for parts/service on the mobile device.
- Updating of current invoice for parts/service on the mobile device.

- Viewing of members previous breakdowns on the mobile device.

### *5.3.3 Further System Processing*

The further processing tasks are as follows:

- Creation of member details on desktop/server.
- Updating of member details on desktop/server.
- Creation of vehicle (members) details on desktop/server.
- Updating of vehicle (members) details on desktop/server.
- Updating of breakdown report on desktop/server.
- Updating of invoice for parts/service on desktop/server.
- Payment transaction on the mobile device.

## **5.4 Data Models**

Using two types of model taken from the SSADM methodology, we can easily represent the data requirements of the system. These are detailed in the next sub sections.

### *5.4.1 Data Flow Diagram*

Data Flow Diagrams are a widely used technique for representing the information flows of a system. The diagrams represent the external agents sending and receiving information; processes that change information; the information flows themselves; and where information is stored. The replacement of the breakdown report store is essential, whilst the replacement of the invoice for parts and service is desirable. See Appendix F section 1 for Data Flow Diagram

### *5.4.2 Logical Data Structure*

This is a method for describing what information should be held by the system. The approach used in SSADM is very similar to entity modelling in other methods; diagrams are produced showing the entities and their relationships. These have been broken down into essential, desirable and further relationships. See Appendix F section 1 for Logical Data Structures.

## **5.5 Functional Requirements**

### *5.5.1 Essential Functional Requirements*

The essential functional requirements are as follows:

- Member details should be viewable on the mobile device.
- Breakdown details should be viewable on the mobile device.
- Members' vehicle details should be viewable on the mobile device.



- As the front-end device will be used on the roadside the time it takes to enter member details, breakdown details, etc must be kept to a minimum.
- There should be minimal typing of data; this may be achieved by pull down menu's, tick boxes etc.
- The method of data entry must be simple and fast.

#### *5.5.2 Desirable Functional Requirements*

The desirable functional requirements are as follows:

- Patrols details should be viewable on the mobile device.
- Parts/service details should be viewable on the mobile device.
- The transfer of data from the front-end device to the desktop/server should be simple and fast so that it can be done by patrols each day with ease.

#### *5.5.3 Further Functional Requirements*

The further functional requirements are as follows:

- The front-end device should cater for some form of member authentication.

### **5.6 Non-functional Requirements**

#### *5.6.1 Essential Non-Functional Requirements*

The essential non-functional requirements are as follows:

- The system needs to be simple and easy to use with clear screens and a standardised design.
- The system should not have a steep learning curve so that it can be implemented easily.
- Data should be held in one place and entered only once.
- The mobile device needs to be inexpensive so that all breakdown companies large or small can use it.
- The mobile device needs to be lightweight so that it can easily be carried by patrols.
- The operating system the mobile device runs on needs to be simple and easy to use.
- The battery life of the mobile device needs to be good with enough life to last at least a full day without a recharge, preferably longer.
- The mobile devices screen needs to be viewable in the dark.

#### *5.5.2 Desirable Non-Functional Requirements*

The desirable non-functional requirements are as follows:

- The mobile device needs to be 'rugged' enough so that it can withstand harsh weather and life in a patrol vehicle.

- The mobile device must have the capability for wireless data transfer.

### *5.5.3 Further Non-Functional Requirements*

The further non-functional requirements are as follows:

- Multiple copies of the breakdown report, parts/service form, and application form should be available through a printout.
- Data should be accessible by other systems through a universal interface.

## 6. Design

---

### 6.1 Technical System Options

This section evaluates some of the available hardware and development options available and then details which are to be used in the project.

#### *6.1.1 Identification of Constraints*

There are several constraints that limit the technical system options; these are mainly concerned with the availability of the tools and devices. Any tools and devices to be used should be freely available either through open source or installed on university machines. Further details of constraints in the project can be found in section 4.1, the feasibility study and in the following subsections.

#### *6.1.2 Front End Hardware Options*

The front-end device is one of the most important factors when looking at mobile computing. In the past the choice was very limited as the field of mobile computing was in its early stages, currently however there is a real influx of new technology as the field begins to thrive and become widespread. Manufacturers keep coming up with ingenious and smarter handheld, wireless devices, mobile phones, personal digital assistants (PDA's) and combinations of any of the above. For enterprises wishing to exploit mobility for business, it is a huge challenge to identify which of these will provide the best solution for them. According to Ken Dulaney, vice president of Gartner (a leading researcher in mobile computing), every six months or so there will be dramatic shifts in device designs and capabilities and in development platforms and this will continue through 2006 (80% probability) [14].

In selecting a suitable device there are a number of issues that affect a devices potential use in businesses. The main factors are the type of hardware, the operating system that runs on the hardware, the user interface and the devices wireless connectivity, these factors are detailed in the next few sections. Different enterprises obviously have different requirements tailored to their needs, the detailed requirements for vehicle recovery companies (specifically using the AA as a basis) can be found in chapter 5.

As there is such a vast choice of devices they have been roughly grouped together for the purpose of this project into the following categories: Tablet computers, laptops, so called smart phones which combine mobile phones with personal digital assistants (PDA's) and handheld computers. These are compared and contrasted in Appendix G. It is worth noting that all of these devices are available in some fashion of rugged device.

### 6.1.2.1 Operating Systems

This section details the different types of operating systems available for mobile computers and some of the advantages and disadvantages of each.

#### *6.1.2.1.1 Palm OS*

The Palm operating system has been the market leader since it was first introduced in 1996. Until recently the Palm OS was seen as the only option for mobile enterprises and businesses. With the introduction of Windows CE based operating systems this has slowly changed. It still leads however in terms of development support and backing with a large back catalogue of both commercial and public software. It therefore has a proven track record of being a platform that can be developed for. The biggest advantages of Palm OS are that each new operating system is backward compatible with the last and that they all have a similar look and feel that is simple and straightforward. This is even true with its use in smart phones, which run Palm OS 5.

Similar to Pocket PC 2002, Palm OS 5 offers wireless connectivity through Bluetooth, IEEE802.11b WiFi (see section 6.1.2.2 for details on wireless connectivity methods) and the mobile phone networks, including wide-area wireless networks. They also offer connection to existing enterprise back-ends such as those developed in Oracle, SQL server etc, via third party software. In terms of development language support see table 1. Palm OS 5 also has 128-bit data encryption and Secure Socket Layer (SSL) support for Internet email, Web browsing, and commercial transactions.

#### *6.1.2.1.2 Windows CE/Pocket PC/Smartphone 2002*

Windows CE was originally released in 1996, but failed to establish itself as a leading mobile operating system as it was littered with bugs and seen as too complex for small devices such as handhelds. Then in 2000 a 'superset' of the Windows CE platform was announced called Pocket PC, which was aimed at high-end handheld users and offered a more simplistic user interface. The latest version Pocket PC 2002 offers basic PDA functions such as memo programs etc, but is orientated more towards the enterprise market. Pocket PC 2002 offers mobile orientated features such as Bluetooth and IEEE802.11b WiFi connectivity as well as access to mobile phone networks. It offers the ability to communicate with other Microsoft products such as office and can have the .NET architecture installed so it can connect to an enterprise's .NET back-end easily. Windows CE derived devices do have major drawbacks in comparison to Palm OS devices in that they use a lot more power and tend to be heavier.

Smartphone 2002 is Microsoft's attempt at entering the mobile phone market. It is a version of Windows CE (as is Pocket PC) however it does not offer touch screen functionality. It also has a more limited development architecture that means it is not as attractive to enterprises as Pocket PC.

### 6.1.2.1.3 Windows XP Tablet edition

Windows XP tablet is a 'superset' of Windows XP Professional and has a familiar user interface; it was released towards the end of 2002. Microsoft applications such as office can be run on the operating system with the introduction of an extension pack that allows the use of the special features. This method can also be applied to applications built by other developers, although any applications have to be altered so that they can interact with the special features of a tablet PC such as the stylus.

Windows XP tablet edition, as Pocket PC and Palm OS, offers handwriting recognition although the software used for this is more advanced. It also, unlike the other operating systems mentioned, offers the ability for speech recognition. A big disadvantage with the operating system at the moment is that it currently only supports IEEE 802.11b WiFi and Bluetooth but no other wireless connectivity methods

	Windows CE Based	Windows XP Tablet Edition	Symbian/EPOC	Palm OS
<i>Java</i>	Yes	Yes	Yes	Yes
<i>C++</i>	-	-	Yes	Yes
<i>C</i>	-	-	-	Yes
<i>Visual Basic</i>	-	-	Yes	Yes
<i>Smalltalk</i>	-	-	-	Yes
<i>Pascal</i>	-	-	-	Yes
<i>Visual J++</i>	-	Yes	-	-
<i>Visual C++</i>	-	Yes	-	-
<i>Visual C#</i>	-	Yes	-	-
<i>Embedded VB</i>	Yes	-	-	-
<i>Embedded C++</i>	Yes	-	-	-

Table 1: The Programming languages available for use with which operating system.

### 6.1.2.1.4 Symbian/EPOC

EPOC was the originally a operating system developed by Psion, however, in mid-1998 Psion joined forces with Ericsson, Nokia and Motorola to form a new joint venture called Symbian. T here aim was to establish EPOC as the de facto operating system for mobile Wireless Information Devices and seeking to drive the convergence of mobile computing and wireless technology. Today Symbian OS v7 supports wireless data transfer through exiting phone networks such as GSM, GPRS and also 3<sup>rd</sup> generation networks. The operating system is more orientated towards the commercial mobile phone market and offers limited support for enterprise and businesses looking to develop data

collection software. It does however support development languages such as Java (refer to table 1) so this may change in the future.

#### 6.1.2.2 Wireless transfer methods

The following section details the current wireless protocols available in the United Kingdom.

##### *6.1.2.2.1 IEEE 802.11b*

802.11b (also referred to as Wi-Fi: Wireless Fidelity) was a 1999 ratification to the original 802.11 standard, allowing wireless functionality comparable to Ethernet. 802.11 specifies an over the air interface between a wireless client and a base station or between two wireless clients. 802.11b provides 11 Mbps transmission (with a fallback to 5.5, 2 and 1 Mbps) in the 2.4 GHz band and is mainly used for wireless LANs. For more detailed information please refer to [18].

##### *6.1.2.2.2 GSM (GLOBAL SYSTEM FOR MOBILE COMMUNICATIONS)*

One of the leading digital systems used for mobile communication. GSM is a 2<sup>nd</sup> generation protocol that uses narrowband TDMA (Time Division Multiple Access) that allows eight simultaneous calls on the same radio frequency. GSM offers bandwidth of up to 9.6 kilobits per second and was first introduced in 1991. The GSM service is available in more than 100 countries and has become the de facto standard in Europe and Asia. For more detailed information please refer to [19].

##### *6.1.2.2.3 GPRS (General Packet Radio Service)*

GPRS is a 2.5-generation protocol is a packet switched technology, based on GSM, which offers 'always on connection' and bandwidth at speeds up to 115 kilobits per second. This however is highly variable depending on the circumstances. Both GPRS and GSM are used in the UK to implement wireless WAN. For more detailed information please refer to [19].

##### *6.1.2.2.4 Bluetooth*

Bluetooth is a short range (between 10-100 metres) radio technology developed by Ericsson and other companies that makes it possible to transmit signals over short distances between telephones, computers and other devices without having to interconnect them with wires. Its primary use is to simplify both communication and synchronization between devices to create wireless personal area networks (WPAN). For more detailed information please refer to [20].

#### 6.1.3 Chosen Front End Hardware

The best choice for an actual system that the AA or other large recovery companies, could use would be based on a tablet device. By using one of these devices instead of either a smart phone or handheld they would have the storage

and processing power for an integrated system (ultimately an integrated system is far beyond the scope of this project) that would offer multiple services, however there would still be some disadvantages. It would still have a shorter battery life than for example handheld devices, would weight much more than handheld devices and would be limited to WLAN connectivity only. Finding a tablet device to actually use for testing would also prove to be very difficult, as they are very new and not yet widespread. This is the same for smart phones; it would be extremely difficult to borrow a device as most people will not have one.

Using a laptop would be the simpler solution, however problems such as the weight and the poor battery fail some essential requirements. Therefore the system will be developed on a handheld device. More specifically a Palm as one can be borrowed for development, from a friend, Andrew Tilley.

Palm devices have been around for some time and there is a strong developer network established. They should be more than adequate for the collection of recovery data and they have room for expansion. As Palm devices are backwards compatible the software can theoretically be ported to a Palm OS smart phone, there is also the possibility to expand the software so that it uses some of the Palm OS 5 features mentioned in section 6.1.2.1.1. For the data transfer Palms also offer both wired and wireless possibilities. Initially implementing transfer over a serial cable through a cradle, there is the possibility to expand to wireless transfer. This however could only be coded and not tested as the device to be used does not have wireless capabilities.

#### *6.1.4 Development Language and Environment Options*

The options available for the development language on Palm OS are as shown in table 1. The choice is quite broad and it includes Java, which is known from the School of Computing Module SO21. One problem is however that PalmSource the owners of the Palm OS currently only officially support C development.

Many of the development environments available, such as Codewarrior, Appforge and Jbuilder Handheld Express require purchasing the software. Having to find free development software reduces the number of possible programming languages dramatically. The free environments available were:

- PRC-Tools: a complete GCC-based compiler tool chain for building Palm OS applications in C or C++.
- Superwaba for Java: Waba is a free open source programming platform that lets developers write one program that can run on Palm OS, Windows CE, or any device that supports Java.
- Pocket Smalltalk: a rapid application development tool for business-quality applications using the Smalltalk programming language.

Pocket Smalltalk was not chosen, as it required learning the Smalltalk language. Superwaba would have been ideal but it is only Java based and not officially certified by Sun. PRC-Tools allows C++ and C development and is

officially supported by PalmSource. The final decision was made after investigating the literature available on the platforms. All the books that were researched are based on the C programming language; there were no books available that dealt with development using Superwaba or Pocket Smalltalk. If one of these options were chosen it would mean total reliance on the languages website and message boards. Therefore the software will be developed in C using the PRC-Tools. As C++ is known from the many School of Computing modules such as SO12, the transfer to C should be relatively simple. This would also allow use of the books available as well as support from the PalmSource developer website. For more details on the development languages and environment available see [21].

## 6.2 Data Design

### 6.2.1 Relational Data Analysis

As the focus of the project has moved more towards producing the software for the front-end device the relationships (Appendix F) will be implemented in stages. To produce an initial prototype, which can be added to later, only the Member, Vehicle and Breakdown entities (see figure 3) will be implemented at first so that the front-end caters for the minimum requirements. The other relationships will be implemented when the additional screens are added.

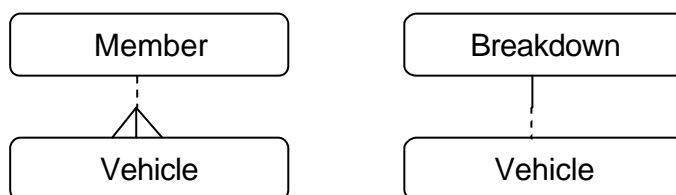


Figure 3: The Member, Vehicle and Breakdown entities and their relationships

**Note:** Due to constraints on the time available the only the initial entities were developed. The other relationships will have to be left until the other screens are implemented (Appendix F).

### 6.2.2 Data Flow Diagram

The new system will improve the existing system by removing the manual stores in the process of breakdown callouts. The new system will have the same flow of data however there will be one computerised data store instead of multiple manual data stores. Refer to Appendix H for data flow diagram of new system.

### 6.2.3 Data Structure

The data structure design went through several phases as the limitations of the handheld device came apparent. The original data structure can be found in Appendix H.



This design was based on the breakdown report (Appendix E) provided by AA patrolman Steve Willingham. There are a few sections from the report that had to be removed, as they would have been too complex to implement on a Palm device, these are such objects as the customer signature.

This design was then changed to allow for variable length strings to save on storage space. Some fields were also removed that were not essential to make the implementation simpler; there was also the need to reduce the size of long strings as they would not fit correctly on screen. The vehicles make and model were also changed from lists to strings, as a full list of all possible vehicles would take too long to implement. The full changes are documented in Appendix H.

After some time programming the system, it became apparent that multiple screens would take much longer than expected to implement. This was due to the large amount of code required to process events from each screen. With this in mind the design was again changed so that it would be simpler to implement and could be completed in the required time. Once again the full changes are documented in Appendix H. The three different entities, Member, Breakdown and Vehicle would now be linked to the same record through an overall structure that wrapped the entities. This is a method taken from the example given in ([22], Chapter 14). Figure 4 shows the basic principal of this technique.

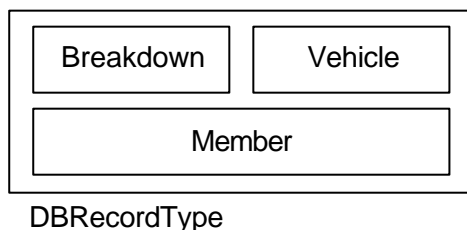


Figure 4: DBRecordType Data structure

### 6.3 Process Design

The processes that occur in the system can be modelled using the SSADM techniques of Entity Life Histories and Entity Event Matrices. Entity Life Histories model the system from a viewpoint of how information is changed. What the Entity Life Histories show is the full set of changes that can possibly occur to the information within the system, together with the context of each change. See Appendix H for the Entity Life History diagram for the system.

The Entity Event Matrix (Appendix H) accompanies Entity Life Histories and shows which events affect which data groups, for more information on these techniques see ([23], Chapter 7).

## 6.4 User Interface Design

The user interface in a handheld device is a very important factor to consider. In Palm OS applications there is a standard format for such objects as buttons and lists forms etc. These are all handled by the Palm OS and simply have to be specified in the code, with their coordinates. There are a number of concepts to consider when designing Palm applications and these are presented in [24]. These include:

- Palms are pocket-sized devices: small screen and no keyboard.
- Applications must therefore limit data entry.
- Menus in Palm applications are hidden and hence their use must be limited.
- Buttons are of a set size which means only a few can fit on a screen at one time. Therefore they should be used only when necessary.
- Less is more: where possible functionality should be moved to the desktop/server.
- Applications should be fast and simple to use. Therefore the number of steps required to perform a task should be kept to a minimum.
- User interfaces should be consistent throughout the application and be similar to other Palm applications.

Taking these factors into account it was decided that the user interface should be similar to those applications that already exist. In the case of this project the nearest application was the Address Book and hence screen designs were modelled on those screens (see figure 5).

The application will be called Recovery; therefore the first screen that appears when the user opens the application is the Recovery List screen. This screen will display all the members on the handheld. The user can simply view one of the members by selecting the members record. This will open up the Member details screen, which will display information on the member. This screen also allows access to the Vehicle details screen through the vehicle buttons and the Breakdown Details screen through the breakdown button. The Done button will return the user to the Recovery List.

The Vehicle Details screen will show the user details of the vehicles the member owns, currently the number of vehicles will be limited to two. The user can also attach a note to a vehicle describing the condition of the vehicle. The Breakdown Details screen will allow the user to enter details about the breakdown. Details will include, the fault that caused the breakdown, what action was taken when the patrol arrived and whether that user has been visited/not-visited. The idea behind the Visited/Not-Visited checkbox is to distinguish between those who have been visited and those who are yet to be visited. This was originally going to be used with the backend of the system. When the user synchronised with the backend a new set of members would be installed onto the handheld, the patrol would then have to visit these members keeping record of which had been visited already.

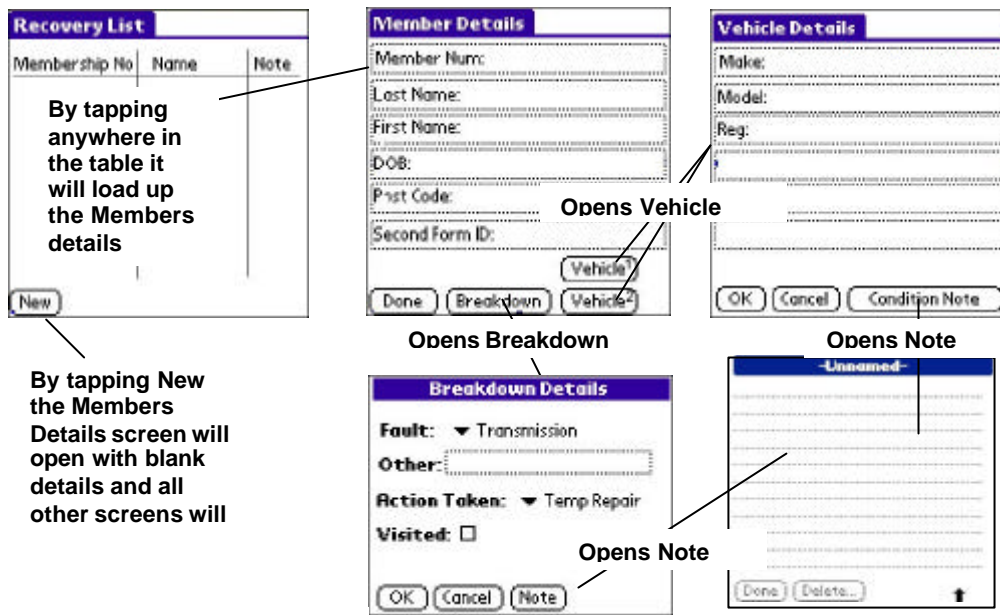


Figure 5: Screen designs

The screen design was changed at a later stage due to difficulties in implementing multiple screens. The amount of coding required for each screen was found to be quite extensive and therefore the time constraints imposed meant the Vehicle Details screen was merged with the Member Details screen. Figure 6 shows the updated Member Details screen.

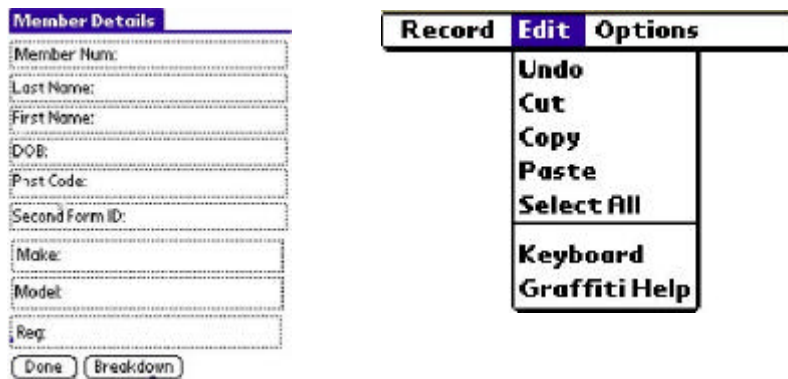


Figure 6: The redesigned Member Details screen, now including vehicle details and Standard Palm OS menu options for editable text.

As the guidelines in [24] state that the use of menus should be kept to a minimum, the menu design for the screens was limited to the functionality that was required. All screens, which require text entry, will have the standard Palm OS text editing menu structure (see figure 6). The only other menu option that will be made available will be one allowing deletion of records. This is only a temporary option and if the full system were to be built this functionality would be removed as patrols cannot remove members records.

## 7. Implementation

---

The purpose of this chapter is to give an overview of the development environment used, outline the stages of development and mention some of the problems and issues that arose. It also describes the methods used to test the software to ensure that it was robust.

### 7.1 Development Environment

The first stage of starting the implementation was to set up the required development environment so that a program could be coded, compiled, tested and debugged. To access the required development tools the Palm OS Developer program was entered into (see [21]). This gave a step-by-step guide on how to set up a development environment for C. The tools used had to be freely available; therefore the ones used were Cygwin, GNU PRC-Tools, the Palm OS SDK, and the Palm OS Emulator. The hardware used was a Palm IIIe kindly donated by a friend. The following sections describe each of these and their uses.

#### 7.1.1 GNU PRC-Tools

GNU PRC-Tools is a set of development tools, created by the open source development community. It contains:

- *M68k GNU C compiler*: a modified version of the GNU C compiler (*gcc*), which transforms source code to Motorola 68000 (the majority of Palm's use this 68k processor) object code.
- *PiIRC*: The PiIRC tool is a resource compiler that transforms text descriptions of a user interface and other resources into the correct binary format expected by the Palm OS. For more details on resource files see section 7.2.2.
- *build-prc*: This utility converts the Motorola 68000 code into the .prc (refer to section 7.2.2 for more details) format expected by the Palm OS, and also combines that code with PiIRC-created resources to produce a finished executable program.
- *gdb*: The GNU debugger is a tool for debugging Palm OS applications at the source code level.

PRC-Tools are the only freely available development tools officially supported by PalmSource, the commercial environment supported is the CodeWarrior package. For more details on GNU PRC-Tools refer to [22]

#### 7.1.2 Cygwin

As the HotSync software provided with the Palm hardware ran under windows it was necessary to install the Cygwin utility so that PRC-Tools could be used in a windows environment. Cygwin is a UNIX emulator for windows that also installs the required files for PRC-Tools. It provides you with a bash shell so that you can run the tools and the compiler.

### 7.1.3 Palm OS SDK v.5

The Palm OS SDK version 5 is a set of libraries, headers and example code for building applications for Palm Powered handhelds. The version used was also specific to PRC-Tools.

### 7.1.4 Palm OS Emulator

The Palm OS Emulator is software that emulates the hardware of the various models of Palm Powered handhelds. It is extremely valuable for writing, testing and debugging (by connecting with PRC-Tools) applications. It creates "virtual" handhelds by running the Emulator with a ROM image. One can download the ROM image, install it and the emulator becomes a copy of the ROM device.

### 7.1.5 Palm IIIe

The Palm hardware used ran on Palm OS v.3.1 and had a monochrome backlit display. It ran on Motorola 16Mhz processor, had 2MB of memory and had TCP/IP and infrared capabilities.

## 7.2 Outline of Development

The following sections provide some detail into the structure of Palm applications and where these structures have been used in the Recovery software produced.

### 7.2.1 Resources used

Many resources had to be used during development as there was a lack of prior knowledge on how to develop for Palm OS. The main resource used was [22], and much of the code used was taken from the examples given in the book. Many of the functions that are used to produce the Recovery software are generic and can be used for many Palm OS applications. These generic functions were downloaded from the books companion website then modified. Other resources used included newsgroups [25], knowledge bases [26], and numerous websites.

### 7.2.2 Palm File Structure

As PRC-Tools were being used to build the software learning the file structure that PRC-Tools requires to produce a fully working application was necessary. A `myApp.prc` file is the final compiled code file similar to an `.exe` file and this is the file that actually gets installed onto the device. To create this file, a `myApp.c` file is required containing all the main code, its appropriate `myApp.h` header file, a `myApp.rcp` resource file (which contains code on how to build the user interface) and its appropriate header file normally called `MyAppRsc.h`.

The resource file specifies structures such as forms, menus etc and the positions of the objects in the structures. For example the following produces some of the Member edit form in Recovery:

```

FORM ID EditForm 0 0 160 160
MENUID EditMenuBar
USABLE
BEGIN
    TITLE "Member Edit"
    TABLE ID EditTable AT (0 18 160 121) ROWS 14 COLUMNS 2
        COLUMNWIDTHS 45 115
    BUTTON "Done" ID EditDoneButton AT (1 147 35 12)
END

```

Each of the variables used in the resource file must be declared in the header for file for the resource. For example the variable `EditForm` used in the above code identifies the form; this would be defined in the header file as:

```
#define EditForm          1100 // all allocated numbers must be different
```

The files used to create the Recovery software are as follows:

- `recovery.c`: main code file
- `recovery.h`: main header file
- `recoveryDB.h`: includes the database structure.
- `recovery.rcp`: contains the code for building the forms, menus, lists, tables, alerts, etc.
- `RecoveryRsc.h`: contains the definitions of the variables used in `recovery.rcp` file.

### 7.2.3 Palm Application Structure

Once it was understood what files were required to create an application the first basic piece of software was written. By doing this, an understanding of how Palm applications were structured was achieved.

Palm applications differ from many other applications, they are totally event driven (refer to Appendix I for diagram showing how events are processed). An event can be user initiated, such as touching the screen with the stylus, the Palm OS can also initiate an event. For example the Palm OS may send an event saying that synchronisation with the desktop is about to occur. The application will do nothing if there are no events to be processed; it will gladly sit there idly waiting until an event occurs. The structure of an application is therefore built around this concept.

All Palm applications will have a main function *PilotMain()*, and this is the entry point for the application. Normally an application will have an initialisation function (in Recovery its called *AppStart()*), which is the first function called in *PilotMain()*. This function sets about opening databases and retrieving the applications saved preferences etc. Once this is completed the application will enter the main event loop (in Recovery its called *AppEventLoop()*), which keeps processing events and sending those events to their appropriate event handlers. When a *StopAppEvent* arises the

application will call its close function (in Recovery its *AppStop()*), which closes any databases, memory handles and saves the application's preferences.

#### 7.2.4 Storage of Data on Palm OS

Once it was established how Palm applications were created, the next step was to think about how the data collected would be stored on the device. The storage of data on a handheld Palm is very different from storing data on your average desktop. There are limitations on the amount of memory and storage available and hence databases as they are normally thought of don't exist. The RAM available for the Palm IIIe used for development is only 2MB. This memory is used both dynamically like the RAM in a desktop and for storage like a hard drive. The dynamic RAM is implemented using a dynamic heap whilst a storage heap implements the storage RAM. Memory chunks in a storage heap are called records. Each record is part of a database that is implemented by the Palm OS data manager. A database is simply a list of memory chunks and some database header information.

All information on the member and breakdown in Recovery is held in a series of structures, with one main structure that references the others. These structures make up the database with each member being a record. Recovery also has a database header, which is known as the application info block. This structure contains information on the labels for fields and generic application information. The full structure can be seen in Design, section 6.2.3.

#### 7.2.5 Data Compression

Due to the limited memory on the Palm IIIe device it was decided that data compression had to be implemented. Recovery's record format is designed to use as little space as possible to store each record. Much of the data stored in Recovery are strings, devoting a fixed amount of space for strings would be wasteful; therefore variable-length strings were used instead. Only enough space to store each string was really required.

To achieve this kind of storage efficiency the same technique employed by the built in Address Book application that comes with all Palm operating systems was used. By doing this two database formats had to be implemented: *RecPackedDBRecord*, a packed format for actual record storage, and *DBRecordType*, an expanded format that is easier to access once a record has been retrieved. In order for Recovery to make use of this dual record structure scheme, two functions were created to translate records between the two formats. The two functions are *PackRecord()* and *UnpackRecord()*. *PackRecord()* takes a record of *DBRecordType* and converts it to a *RecPackedDBRecord*. This is done by taking all the strings in *DBRecordType* and compacting them all into one null delimited string, which is placed in *RecPackedDBRecord*. *UnpackRecord()* simply takes the null delimited string and iterates over the pile of strings placing each string in the appropriate field in the *DBRecordType*. A full explanation of each structure can be seen in Design, section 6.2.3.

### 7.2.6 Drawing Functions

Much of the drawing in a Palm application is taken care of by the resource file which when compiled will produce buttons, tables, etc in the standard Palm format. Drawing changing data or altering the user interface however is quite complex and requires the use of the Palm API, in particular the window functions. For further details on window functions see ([22], Chapter 10).

Drawing functions are used in both the Recovery List window and the Member Edit window. On the Recovery List window members' names and their membership numbers have to be displayed. To display the data Recovery uses the `WinDrawChars()` API function which is wrapped in `DrawCharsInWidth()` function. The `DrawCharsInWidth()` function returns the passed in string with concatenated dots if it exceeds the length of the column in the table (refer to figure 9). Like many of the windows functions `WinDrawChars` takes four arguments: a pointer to the characters to draw, the length of the characters in bytes and the x and y coordinates of where the characters should appear.

### 7.2.7 Categories

Every record in a Palm OS database has an attribute field, which contains half a byte of category information about the record. This mechanism allows the user to have customisable categories for classifying records. Although this is not mentioned in the original design the categories were added as they are used in almost all applications, including the built in Address Book, To Do List, and Memo Pad applications. The code was simply taken from the Address Book application and modified. This was quite easy as the Palm OS provides a category manager to make managing implementing them easier.

In order to provide some default categories in the Recovery application, the category names are stored in the resource file `recovery.rcp` as strings. These are then initialised into the application info block when the application starts and runs `AppStart()`. Recovery has default categories "100 Roadside", "200 + Relay", "300 + Homestart" and "400 + Relay Plus" (see figure 8). These represent the membership types of the AA. The categories can be changed in the Recovery List, Edit and Breakdown Details screens.

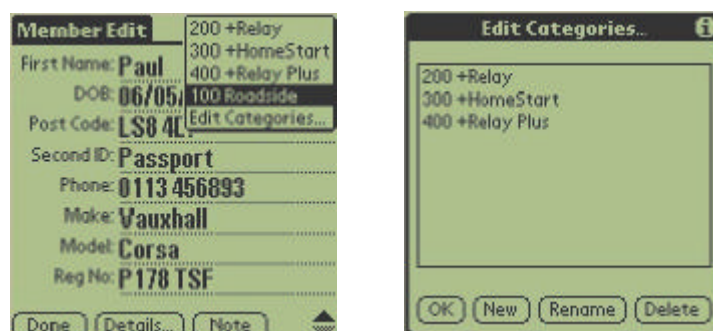


Figure 8: The membership categories displayed in the Member Edit screen and the Edit Categories screen accessible by selecting Edit Categories...



### 7.3 Recovery Screens and Menus

Once the underlying functions had been implemented such as the creating, writing and reading from the database, the next phase was to produce the screens of the Recovery system. In the design it was decided to follow similar design rules as those of the built in Palm applications such as the Address Book to ensure that the system had a similar look and feel to other Palm applications (refer to section 6.4 for more details). The following sections outline Recovery's screens and detail how each one works and which functions are called by which screens.

#### 7.3.1 Recovery List

When the user taps the Recovery icon (refer to figure 9) it starts the application and the Palm OS calls the *AppStart()* function which then calls the *FrmGotoForm()* API. This API is used throughout the application; it generates an event that will open a form. This event is processed by the main event handler *AppHandleEvent()*, which passes the event to the corresponding forms event handler, in this case *ListFormHandleEvent()*. All the forms used in Recovery that require user interaction processing need an event handler. This event handler deals with all possible events that can occur within that form, this includes the form being opened and closed.

Most of the Recovery List screen is taken up by a scrollable table that links to the Recovery database and loads each members record. It displays their membership number, last name, first name, whether they have been visited or not and if they have an attached note. This task although sounding relatively simple was one of the most difficult. The table has to be initialised first before it will interact with the user and this process is done in *ListFormInit()*. All table columns need to have data types assigned to them and then made usable. In the case of the Recovery List form the data type was actually a custom type as the data had to be loaded from the database and the columns of the record had to process stylus information, rather than simply entered by the user. With a custom data type the drawing of the data has to be handled by a call back function *ListFormDrawRecord()*. This call back function draws the simpler elements in the Recovery List such as the note icon, however it uses the helper function *DrawRecordName()* to retrieve the members name and membership number from the record and then draw those, for more details on exactly how drawing occurs see ([3], Chapter 10).



Figure 9: The Palm IIIe and its main screen and the Recovery List screen in the Recovery application.

Once the drawing call backs have been set the records from the database are actually loaded into the table using *ListFormLoadTable()* which in turn are drawn by the call back function (refer to figure 9).

Now that the records were displayable a method of creating a new record, was needed. Previously for testing purposes records were simply created within the code. Now a new blank record would have to be created and then appended to the database. The new button at the bottom of the Recovery List screen creates a new member and opens the Member Edit screen (see section 7.3.2). To create a new member it calls the *EditFormNewRecord()* function which creates a new record in memory, fills the record with nulls and then attaches that piece of memory to the Recovery database.

For the user to open an existing record they had to have a method of selecting a member, which opens the Member Edit screen and loads the screen with the member data. This could have been done using a button, however it was decided that Recovery should behave like similar Palm applications and allow the tapping of a record in the Recovery List screen to open the Member Edit screen. All tables in Palm OS have the ability to raise events that notify the application that that particular column and record has the focus. The *ListFormItemSelected()* function handles these events using a case statement. Depending on the column selected the *ListFormItemSelected()* function will either go to the Member Edit screen (if the column selected is either the name or membership number), open the note screen, or display the selection pop up for visited/not-visited. The global variable *gEditRowIDWhichHadFocus* keeps track of which is the current record selected and is used by all of the functions called in *ListFormItemSelected()* to load data or to associate a something such as a note to a particular member.

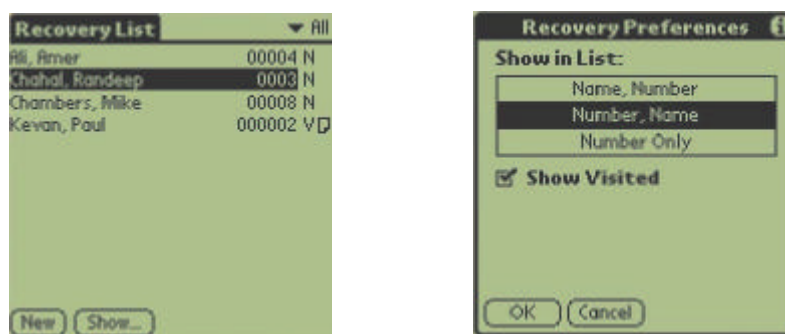


Figure 10: The Recovery List sorted by last name and the Preferences screen.

So that the user can order the Recovery List in the order they wanted the method explained in ([3], Chapter 14) was used. This was based on some of the code given in the book that used a global variable called *gShowInList* to adjust the list dimensions, the column order and change the sort order of the database. This global variable is passed into a function called *RecChangeSortOrder()* which uses the *DmQuickSort()* API to sort the database.

The global variable `gShowInList` can be changed in the Recovery Preferences dialogue which can be displayed from the Recovery List screen by tapping on the show button (see figure 10). This also allows the user to select whether they want to display the visited/not-visited column by ticking the box; this again sets a global variable called `gShowVisitedStatus`. The global variables are used in a case statement in the `ListFormDrawRecord()` function which determines which way the list should be drawn.

### 7.3.2 Member Edit

The Member Edit screen is where the user can input and change the data belonging to a member (see figure 11). This screen is opened when the user selects the new button or when the user taps on an existing member record in the Recovery List screen. Like the Recovery List form, Member Edit also has an event handler `EditFormHandleEvent()`, an initialisation function `EditFormInit()`, and is mainly made up of a table. In this case however the table is made up of two columns, one which holds text labels and the other which allows the user to input text. Each row in the table represents a text field in the Recovery database (see section 6.2.3), with its appropriate label (string labels are declared in `recovery.rcp`). The table however uses standard column types of `labelTableItem` and `textTableItem` so that the Palm OS deals with the drawing of the characters without the need of custom draw back functions. This can be done here, as the fields are editable and don't launch other windows as the list table in the Recovery List form.

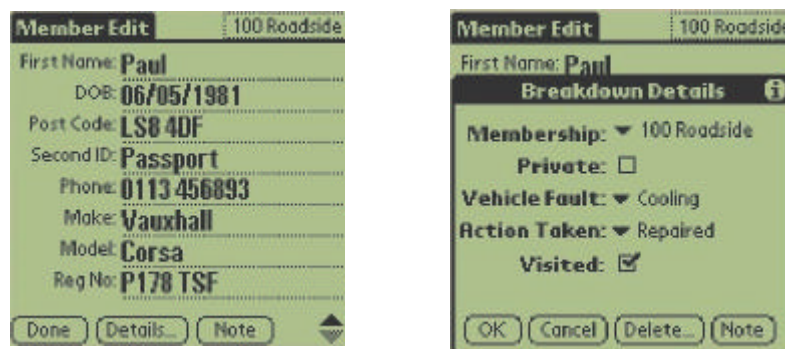


Figure 11: The Member Edit form and Breakdown Details pop up form.

If the Member Edit form is opened using the new button on the Recovery List form then the text fields are left blank as the new record is filled with nulls. If the form is opened from the user selecting a record in the Recovery list then the text fields are filled with the details of that record. To deal with this and the saving of data entered into the Member Edit form two table call back functions were created. `EditFormGetRecordField()` returns a pointer to the current field of the record that was selected and `EditFormSaveRecordField()` saves the current text field. The `EditFormGetRecordField()` function is used by the `EditFormLoadTable()` function to load the fields within the table with the fields from the database. The `EditFormSaveRecordField()` function however is used by the `EditFormSaveRecord()`

function to save a record, this function also checks to see if there has been text entered in any of the fields in the table. If there has not then it will not save the record. This was done in case the user accidentally creates a new member and still selects the done button instead of cancel. The records are saved and loaded per field so that any changes in field does not require the entire record to be saved merely those that have been altered. The changes in these fields are tracked using the `RecDBRecordFlags` structure, which holds Boolean values for each of the fields.

The two buttons at the bottom of the screen, details and note, allow breakdown details or a note to be attached to the current record. The details button simply creates an event that opens the details form whilst the note button will call the function `CreateNote()` first to ensure that a note is created in the record before opening the form screen.

### 7.3.3 Breakdown Details

The manual breakdown report used in the design was far too complex for all the details to be presented on the Palm device, see section 6.4 for more details. Therefore breakdown details were kept to a minimum, which lead to the production of the Breakdown Details form that allows the user to change the breakdown status types, which are declared in the `recoveryDB.h` file. The breakdown status types are as follows: the actual fault leading to the breakdown (`faultStatus`), the action taken to solve the fault (`actionStatus`) and whether the member has been visited yet or not (`visitedStatus`). The basic layout of the form (see section 6.4) was taken from the built in Address Book application and from the code given in [22].

The Breakdown Details form is much simpler than the previous forms as it does not require the use of tables. Much of the functionality such as the pop up lists are handled by the Palm OS and specified in the `recovery.rcp` file. The strings to be used by the lists and the position of the pop up trigger (which launches the list) are simply specified in this file. The application then gets the current member record using the `RecGetRecord()` function and sets the lists to display the status values held in the rec ord. This is done through the `DetailsFormInit()` function. If the record is new then the lists are set as default.

Apart from the statuses there are two other options that can be changed in the form these are the built in Palm OS options, category (see section 7.2.7) and private record. The category used in the Recovery application is the membership type so the user can change a member's membership type if they need to. Creating private records is a standard function on all Palm OS after 3.0, which allows the user to hide records by switching on the hide private records option in the OS. The code for this is generic and taken from [22]; it simply sets a flag in the record.

The buttons at the bottom of the form allow the user to cancel any changes, save the changes made, delete the record, or add a note by opening up the note screen. Changes will only be saved by the `DetailsFormSave()` function if the values in the lists differ from those held in the status fields in the record. The record deletion button will display a

dialogue that allows the user to delete a record it also allows the user to specify that the record should only be removed from the handheld and a backup should be placed on the desktop PC. This again is a standard method taken from [22].

#### 7.3.4 Note Form

The note form in Recovery was needed so that patrols could enter any extra information regarding the breakdown that they needed (see figure 12). This was essential, as the Recovery software did not have the full functionality of the manual breakdown form (the missing functionality can however be added later if needed). The Note form was based on the note form used by the built in application Address Book so that implementing it would not take a large amount of time. With small modifications such as displaying the members name at the top of the form, the note form was quickly added.

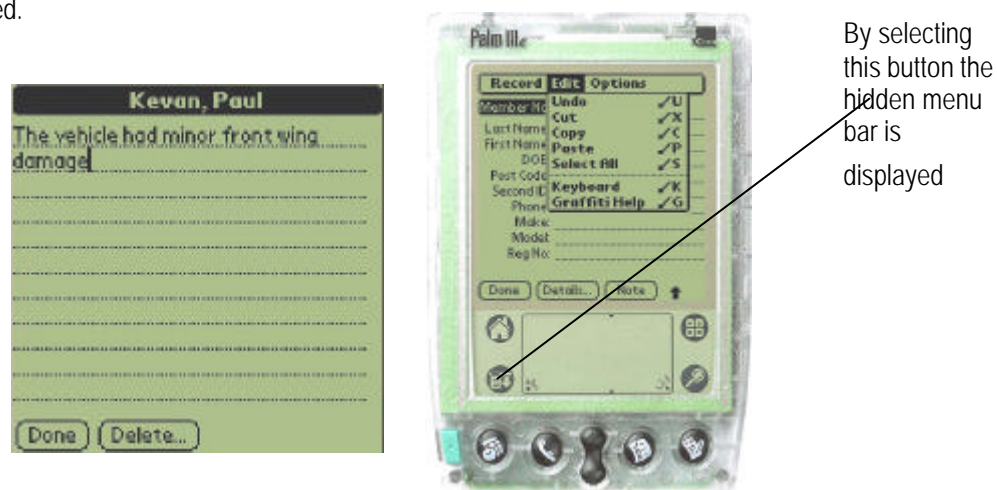


Figure 12: The Note Form in Recovery and the Member Edit screens hidden menu.

#### 7.3.5 Menus

The menu's in Recovery were fairly simple to implement as they were kept to a minimum in the design see section 6.4. The selection of a menu generates an event that is caught by the active forms event handler. For an example the Member Edit screen has a event handler called *EditFormHandleEvent()*, which will process the menu event when a menu selection is made (see figure 12) whilst that form is active. *EditFormHandleEvent()* will then pass the event onto the appropriate menu handler. The Member Edit screen menu is handled by the *HandleCommonMenus()* function which was used so that the standard menus could be used in all text editing screens. This then calls the appropriate functions, for standard features such as cut and paste calls the Palm OS API, *FldCut()* *FldPaste()* respectively. These API take the current field highlighted as their argument and the Palm OS handles everything else.

The only other menu options implemented apart from the standard text editing functions, was the deletion of the current record and the about option. The delete record option would just call the function *DeleteRecord()* which deletes the current record and the about option shows the details of Recovery. These menus can be added to in the future so that they offer additional functionality such as infrared beaming of member records, this can be left as a possible enhancement.

#### 7.4 Problems and Issues

- The first problem not anticipated was the actual length of the coding task. Recovery was developed on a Palm OS handheld as one could be accessed easily. It was programmed in C due to the available support from the Palm developers website and the existing Palm programming books. However only after reading through [22] and building the 'Hello World' program did it become apparent that the time set to produce the complete system was too little. This was resolved by shifting the project scope to be more handheld focused and research based, so that there was more allocated time for the handheld device.
- From Palm OS 3.5 onwards the operating system would use different data types than those used before. This would not change backward compatibility for the older Palm operating systems as later SDK's included header files that dealt with the conversion. Therefore the text book being used for development was changed from [27] to [22].
- The number of screens in the system could have been improved on. If the time had permitted it would have been better to have one screen for the member details and one for the vehicle details so that multiple cars could have been allowed. The relationship is now only one to one (see figure 13); this was however necessary to complete the system in the required time, the extra screens can be implemented at some point in the future.
- It was originally perceived that the front end would connect to a server so that data would be transferred. This to an extent has occurred; an actual conduit (the software for handling the data transfer) has not been implemented however the generic conduit handles the backup. Recovery's database backup bit which tells the conduit to back it up is set in the *RecGetDatabase()* function which is called when the application opens. A purpose built conduit would be better as it would allow data manipulation and checking to occur. However after investigating the requirements that this would take it was decided that the time did not permit this feature. Creating a conduit requires a completely different Conduit SDK, a different development environment and language, Visual C++ or Java and requires code to interface with the handheld devices cradle. This is a large task in its self and will have to be a possible extension.

- Other minor problems that could be improved include data type masks that insisted in a set format. It would be better for name fields to be capitalised automatically and the date of birth field to have a specific type.

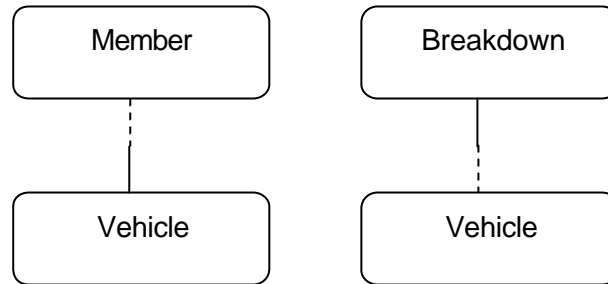


Figure 13: The relationships between entities as they currently stand.

## 8. Evaluation

---

The purpose of this chapter is to provide details about the three different types of testing approaches used and analyse the choices made throughout the project. It will also assess the advantages of the tools used and evaluate the new system produced. Finally it will evaluate the future of mobile computing and state areas of further study and development.

### 8.1 Testing

#### 8.1.1 White box testing

The system was tested in units following the spiral model. This was done so that any problems could be reviewed and then if changes needed to be made they could be. This unit testing utilises the white box testing technique (refer to [28] for more details), which incorporates:

- Testing that all *independent paths* within a module have been exercised at least once.
- Examining all logical decisions on their *true* and *false* sides.
- Executing all loops and testing their operation at their limits.
- Exercising internal data structures to assure their validity.

To help in this testing I used a special function of the Palm emulator called Gremlins. The tool allows you to create objects called gremlins, which test your specific application. Each Gremlin has the following characteristics:

- It generates a unique, random sequence of stylus and key input events to step through the user interface possibilities of an application.
- It has a unique "seed" value between 0 and 999
- It generates the same sequence of random events whenever it is run.
- It runs with a specific application or applications.
- It displays a report immediately when an error occurs.

By running a number of Gremlins over a period of up to one hour I could with a fair amount of certainty say that all functions, buttons, menus etc in that unit were tested (Appendix J contains an example gremlins log). This testing produced some errors regarding display parameters and unlocked memory handles. Most problems were resolved, however the application occasionally has a memory leak when the Gremlins are run over a long period of time (more than one hour). This is due to a memory handle staying locked when the application terminates. Some time was spent trying to find the cause of the memory leak, however as it only occurred very occasionally the exact cause was not detected. Although this is unwanted, memory leaks do not effect the application, as it will not even notify the user that



a leak has occurred, only in the emulator does it notify the user. This is because all handles are released when the Palm is turned off; therefore to an extent they are tolerated. If time permitted then it would have been better to find the cause and resolve it, however with the time constraints and that the error was not serious it was left as a future enhancement.

### *8.1.2 Black Box Testing*

Black box testing is designed to test the functionality of the application and uncover any omissions and deficiencies in the way that it works. This was carried out once implementation was complete and covered the following areas:

- Creation/editing/removal of member: To ensure that members could be created then edited and finally deleted.
- Breakdown details selection: Testing whether all the possible combinations of breakdown type and action are selected.
- Alteration to membership type: Ensuring that membership types for members could be changed if needed.
- Creation/editing/removal of notes: Testing that all possible methods of attaching a note worked correctly and that the note was attached and deleted from the correct member.
- Ordering of Recovery List screen: Ensuring that the sorting of the Recovery List screen worked for all cases.

A number of trivial problems arose from the testing mainly to do with the displaying of objects on forms. Some labels were found to be too large for the screen and interfered with other object altering their values. Also some repeated changes required more memory than they were allocated. These have all now been resolved.

### *8.1.3 User Acceptance Testing*

User acceptance testing is the process of getting actual users to test the application. This proved particularly difficult as the potential user who was to test the application was in another city; this meant that I had to post the Palm device via special delivery to Mr Willingham for him to test. As there was limited time, I had to send a prototype version of the software, which had the menu and sorting functionality missing. I explained to Mr Willingham that this would be the case and he accepted that some parts of the application would be completed at a later stage.

A phone interview was arranged with Mr Willingham for the 13<sup>th</sup> of April 2003, so that feedback on the system could be received. The interview took place over the phone as Mr Willingham needed to have the system explained to him as there was no user manual at that time (Appendix K is the write up to the interview and contains the questions asked). The questions asked tried to evaluate not only the application produced (Recovery) but also the platform (Palm OS) it was provided on. His comments can be summarised as follows:

Platform positives:

---

- "It's a lot simpler than Windows"
- Using the stylus is easier than using a mouse or touch pad out on patrol.
- The device was light and small.

Platform negatives:

- Stylus would easily be misplaced in the patrol cab.
- Data entry without the foldable keyboard is a little awkward.
- The device is too limited for the needs of a large corporation such as the AA.

Application positives:

- Very straight forward and simple to use.
- Much simpler than the manual breakdown form.
- Has a similar look and feel to other applications on the handheld.
- The amount of data input is reduced as member information is already there.
- Small learning curve.
- Well suited to simple data collection.
- With the addition of wireless connectivity and communication with the AA's backend systems would save time and effort transferring information.
- Removes the 'paper trail'.

Application negatives:

- Lacked some of the functionality of the existing breakdown form.
- Would require a fair amount of work and additions to make it a viable solution for the AA.
- An integrated system that could incorporate their existing front-end systems would be better.

It was clear from some of the answers given by Mr Willingham that his outlook of what should be included in the systems was based on user (patrol) preferences rather than the business needs. In general the feedback was positive for the application produced although it seemed that Mr Willingham was expecting a much larger and complex system.

## 8.2 Future Improvements

There are a number of improvements that can be made to the system in the future, the main ones are as follows:

- Implement another vehicle screen to allow the user to have more than one vehicle.
- Add more advanced objects from the breakdown report
- Add the additional invoice/parts and service screen to the handheld.

- Write a conduit for the handheld so that data manipulation can occur.
- Develop a backend to centrally store all the information gathered.
- Develop a wireless method of transferring the front-end data to the backend device.
- Create some sort of interface with a printer so that reports can be printed.

Some of these improvements however would take a long time to implement, as they are quite complex. These may be suitable as actual projects in their own right.

### **8.3 Evaluation of Chosen methodology**

One of the first decisions made was which methodology should be adopted. Selecting the appropriate methodology would direct the project by setting out guidelines etc. To assess this choice the following criteria were used:

- Was the methodology suitable for the project? – SSADM can be adapted to any project. However it was found that many of the stages of SSADM required a fair amount of work even when using a reduced version. It became apparent that the methodology caters more towards the collection and analysis of background data and the design. It offered little help in managing the implementation and in testing, for these extra research and other tools had to be used (see section 8.1)
- Were the methodologies tools appropriate? – As the existing AA system is mainly based on data flow, the techniques used (see section 6.2) were very helpful in understanding the actual system and the flows of data.
- Was the process model used by the methodology suitable? – The process model used was the spiral model. This was very helpful, as it requires each stage to be reviewed, this encouraged me to test each section implemented before moving on to the next.

In conclusion the methodology was suitable to an extent, it would be more suitable towards a project, which has more time for detailed documentation. On reflection a simpler technique such as RAD may have been better and allowed more implementation to have been produced.

### **8.4 Evaluation of Front End Hardware**

As mentioned in section 6.1.2 the hardware used for the Recovery software was selected due to constraints on the availability of other options. This consequently impacted the software produced and the devices appropriateness towards the requirements of vehicle recovery companies such as the AA. To evaluate the Palm IIIe device the non-functional hardware requirements for the front-end device (see section 5.6) were used as criteria. These were broken into essential and desirable requirements.

Essential Front End Hardware requirements:

- The mobile device needs to be inexpensive so that all breakdown companies large or small can use it. – Palm IIIe is inexpensive approximately £100.
- The mobile device needs to be lightweight so that it can easily be carried by patrols. – Device is very lightweight and can easily be held in your hand.
- The operating system the mobile device runs on needs to be simple and easy to use. – Mr Willingham the potential user commented on how simple the interface was (see section 8.1).
- The battery life of the mobile device needs to be good with enough life to last at least a full day without a recharge, preferably longer. – Device lasts many days on three AAA batteries.
- The mobile devices screen needs to be viewable in the dark. – Device has a backlight.

Desirable Front End Hardware requirements:

- The mobile device needs to be 'rugged' enough so that it can withstand harsh weather and life in a patrol vehicle. – Although the device is not 'rugged' other Palm OS devices are available that offer this requirement, see [12] for details.
- The mobile device must have the capability for wireless data transfer. – Although the device used does not have wireless capabilities other Palm devices do provide access to GPRS and IEEE802.11b, see [12] for full device specifications.

Although the device fulfilled most of these criteria, some crucial criteria were not included in the original requirements. These included processing power, memory and storage on the device, in these requirement the device faired badly. The potential user also had issues with the device; Mr Willingham found it difficult to enter data using the stylus or displayable keyboard (section 8.1 for more details).

Another problem found when developing for the device was that its operating system limited the functionality of the software produced. Although the simple operating system is an advantage for the user, it is the big disadvantage for the developer as even simple tasks take large amounts of coding to complete.

In conclusion it may have been more suitable to use a tablet PC, which would provide a better platform for development and functionality. Obtaining a device however would still have been a major problem.

## **8.5 Evaluation of Development Language**

As the development environment had to be freely available the choice of language was limited (see section 6.1.4). The choice made was to an extent the correct one, it may possibly have been better to select Superwaba the Java

based language instead, as Java is known from the School of Computing module SO21. After some further investigation it was found that using Superwaba would have reduced the amount of code needed to write the application, although the problems of Superwaba having limited support, no development environment or debugger, and it not being certified by Sun Microsystems would still have been factors. It would have been preferable to build the application in one of the commercial development environments that support Java such as IBM Websphere Studio Device Developer, but even then there are still no textbooks available on developing Java applications for the Palm OS. In conclusion out of the options that were available C was the correct choice made.

### **8.6 Evaluation of Development Environment**

Out of the possible development environments that could have been used (see section 6.1.4), the correct choice was made. The main reasons for this conclusion is that other environments did not offer debuggers and some did not even offer development environments. A debugger is essential when developing Palm software, as many of the errors the emulator generates are difficult to understand. For example a common error would be referencing memory in the incorrect location, without the debugger this would be extremely difficult to locate.

### **8.7 Evaluation of System**

To evaluate the Recovery software produced, it is necessary to compare the system with the existing manual system and other systems implemented. It is also essential to judge the system against the minimum requirements (see section 1.4), the requirements specification (see chapter 5) and finally evaluate the comments made by the potential user (see Appendix K). To evaluate all of these the following criteria were used:

- Did the system meet the minimum requirements? Exceed them?
- Did the system solve the weaknesses found in the AA's existing system?
- Did it fulfil the essential requirements? Desirable requirements? Further requirements?
- Was the system to the approval of the potential user?
- How does it compare to other existing computerised systems used in work field environments?

*Did the system meet the minimum requirements? Exceed them?*

The minimum requirements of the project are as follows:

- The front end should be portable and have the ability to capture and store information such as basic vehicle details, basic member details and simple repair notes that are input by patrol vehicles when they respond to a call. – Recovery is extremely portable as it runs on a Palm device, it allows the user to capture and store, the details mentioned and also allows the entry of breakdown details.
- This information should then be transferred from the hand-held device to a desktop PC via a physical cable, for example a docking station. – Recovery uses the Palm OS cradle, which connects to a desktop PC

through a serial cable. Using the HotSync desktop software's built in conduit it backs up the Recovery database.

Therefore the minimum requirements were met, they were not exceeded however. Due to the lack of understanding of Palm development at the start of the project the extra functionality and features of the system could not be implemented in the appropriate time. They will now have to be left as future enhancements.

*Did the system solve the weaknesses found in the AA's existing system?*

All the problems specified in section 4.3.1.2 were solved apart from one; this was the problem of not having all the data held centrally. After the shift in project scope the backend system originally devised was not developed therefore storage of all data could not be held centralised. The data collected by the handheld is however held centrally on the handheld

*Did it fulfil the essential requirements? Desirable requirements? Further requirements?*

All of the essential requirements were met, however due to time constraints and the shift in project scope the desirable and further requirements were not met.

*Was the system to the approval of the potential user?*

For the simple purpose of data collection and the removal of the Breakdown Report, Mr Willingham agreed the system was good and achieved what the basics required. However it was clear from some of the answers given by Mr Willingham (see Appendix K) that he was disappointed with the lack of additional functionality that the system offered and that he was expecting a much larger, integrated system. This was due more to the actual hardware used, the lack of understanding of Palm development and poor time estimation. It was concluded that the system produced would be a suitable basis for smaller vehicle recovery companies and companies which require simple data collection. The AA however would still require a full-integrated system, which was not the scope of this project.

*How does it compare to other existing computerised systems used in work field environments?*

In comparison to such systems as eTrace (see section 4.3.3.1) Recovery compares well to the front end of that system. It offers similar electronic clipboard like capabilities for data collection. It does not however have a wireless method of transferring data although this could be added later. In comparison to other systems such as OmniExpress (see section 4.3.3.2) and the RAC's patrol PC (see section 4.3.2) it will not compare well as these are large advanced systems and not in the same category as Recovery.

In conclusion the system produced fulfils the needs of basic data capture, storage and manipulation. By doing this it has fulfilled the minimum requirements, solved most of the AA's current problems with data collection and fulfilled the essential requirements specified. After some expansion it would be suitable for use by smaller vehicle recovery companies and could be adapted for other businesses that require data collection in the work field environment. If the AA only required mere data collection and storage then it would also be suitable for them, however they require more advanced systems and therefore it would not be suitable. This is due to the actual Palm devices limitations, which in turn limit the Recovery software.

### **8.8 Evaluation of overall project**

To assess the overall project it is necessary to check that the initial objectives set out have been met. The projects objectives were as follows:

- To research the current manual processes used on the road by patrols and explore the problems they face.  
– This research was completed and a list of problems was compiled.
- To research and explore the possible hardware and software solutions to data capture on mobile computers.  
– Extensive research has been performed and the results of this have been input into the report.
- To implement a front end solution to replace the manual processes. – The Recovery software was produced which replaces the manual Breakdown Report.
- To evaluate the solution and review how it improves the current processes. – The solution has been extensively evaluated against various criteria.
- To review how the solution could be extended. – A list of possible enhancements is provided in section ????
- To reflect on the future of mobile computer technology and what possible applications wireless devices could be used for. – A evaluation on the future of mobile computing is provided in section ????

Therefore the project achieved its objectives and provided a detailed investigation into the field of mobile computing.

### **8.9 Evaluation of the Future of Mobile Computing**

With the influx of wireless technologies and mobile computers it seems that the time has come for the wireless revolution. Currently all the big players in the computing world are joining the rush to add wireless technologies to their products and software to enable mobile system. Due to the technology market slumping after the Dot.com bubble burst; many development companies are seeing mobile computing as the only way to generate income, whilst businesses see mobile computing a method of streamlining their systems to reduce costs.

There are however still major problems that need to be resolved before wireless computing will fully be utilised by businesses and consumers. The new mobile devices such as the tablet PC and handhelds have now come to a point where they offer the storage and processing that can replace conventional PC's yet they do not have the wireless infrastructure to operate successful wireless wide area networks. Current wireless protocol systems such as GPRS simply do not offer the bandwidth required (see section 6.1.2). For the real revolution to start we shall have to wait for the release of third generation phone networks which will then provide the desirable bandwidth for affordable fast work field systems for mobile employees.

From the research carried out for this project, the present future of mobile computing lies in the adoption of IEEE802.11b, which is already starting to change the way the office operates by changing LAN's into wireless LAN's. Looking past that the introduction of 3G will inevitably cause wide adoption of mobile devices and the movement from wireless LAN to wireless WAN. The time shall come when wires become a thing of the past.

#### **8.10 Conclusions**

The project has met its objectives and met the minimum requirements and therefore has been to an extent successful. After some additional work, smaller vehicle recovery companies or any company that requires information to be collected by mobile employees could use the software produced. The software and report produced could also be used as a platform for future development or a basis for research into the field of mobile computing.



## References

- [1] JING J, HELAL A, and ELMAGARMID A (1999): *Client Server Computing in Mobile Environments*, **ACM Computing Surveys**, Vol. 31, No. 2, June 1999, Chapter 1.
- [2] ZASLAVSKY A and TARI Z (1998): *Mobile Computing: Overview and Current Status*, **School of Computer Science and Software Engineering, Monash University**. URL: <http://goanna.cs.rmit.edu.au/~zahirt/Papers/acj.pdf> [30th April 2003]
- [3] The Software and Information Industry Association (2001): *Building the Net: Trends Report 2001 Trends Shaping the Digital Economy*. URL: <http://www.trendsreport.net/wireless/1.html> [30th April 2003]
- [4] FORMAN G.H. and ZAHORJAN J (1994): *The Challenges of Mobile Computing*, **IEEE Computer**, Vol. 17(4): 38-47
- [5] AA Breakdown Recovery Home Page, URL: <http://www.theaa.co.uk/breakdowncover/join/brejoin001.asp> [30th April 2003]
- [6] RAC Breakdown Recovery Home Page, URL: <http://www.rac.co.uk/breakdowncover/> [30th April 2003]
- [7] Green Flag Breakdown Recovery Home Page, URL: <http://www.greenflag.co.uk/breakdown/index.html> [30th April 2003]
- [8] HUGHES B, COTTERELL M (1999): **Software Project Management, Second Edition**. McGraw-Hill.
- [9] AVISON D, FITZGERALD G (1995): **Information Systems Development, Second Edition**. McGraw-Hill.
-

- [10] Gearworks: eTrace Home Page, URL: <http://www.gearworks.com/products/etrace.cfm> [30th April 2003]
- [11] Qualcomm Incorporated: OmniExpress Home Page, URL: <http://www.qualcomm.com/qwbs/products/omniexpress/> [30th April 2003]
- [12] Symbol Technologies (2003): Mobile Palm SPT1800 Home Page, URL: [http://www.symbol.com/products/mobile\\_computers/mobile\\_palm\\_spt1800.html](http://www.symbol.com/products/mobile_computers/mobile_palm_spt1800.html) [30th April 2003]
- [13] At Road Inc (2001): Home Page, URL: [http://www.atroad.com/usa/productsandservices/products\\_othersol.htm](http://www.atroad.com/usa/productsandservices/products_othersol.htm) [30th April 2003]
- [14] GREEN-ARMYTAGE J (2002): *Choosing Mobile Applications*, **Symposium/ITxpo 2002 News Coverage**. URL: <http://symposium.gartner.com/story.php.id.2918.s.5.html> [30th April 2003]
- [15] Sony Ericsson: P800 homepage, URL: <http://www.sonyericsson.com/P800/main.htm> [30th April 2003]
- [16] Electrovaya Home Page, URL: <http://www.electrovaya.com/> [30th April 2003]
- [17] Compaq: Evo 800 Home Page, URL: <http://h41102.www4.hp.com/products/notebooks/evo/n800c/index.html> [30th April 2003]
- [18] IEEE 802.11 Main General Info Page, URL: <http://grouper.ieee.org/groups/802/11/main.html> [30th April 2003]
- [19] GSM World, URL: <http://www.gsmworld.com/index.shtml> [30th April 2003]
- [20] Optimising Bluetooth Bandwidth, Online Library for Bluetooth Developers, URL: <http://www.csr.com/enews/sw007.html> [30th April 2003]
- [21] Palm OS Tools and Downloads, URL: <http://www.palmos.com/dev/tools/> [30th April 2003]

[22] FOSTER L.R. (2002): **Palm OS Programming Bible, 2<sup>nd</sup> Edition**. Wiley Publishing Inc.

[23] ASHWORTH C, GOODLAND M (1990): **SSADM, a Practical Approach**. McGraw-Hill Book Company.

[24] OSTREM JEAN (2002): Palm OS User Interface Guidelines. URL:  
<http://www.palmos.com/dev/tech/docs/> [30th April 2003]

[25] Google Groups: Pilot.programmer, URL:  
<http://groups.google.com/groups?hl=en&lr=&ie=UTF-8&group=pilot.programmer> [30th April 2003]

[26] PalmSource Developer Knowledge Base, URL: [http://kb.palmsource.com/cgi-bin/palmsource.cfg/php/enduser/std\\_alp.php](http://kb.palmsource.com/cgi-bin/palmsource.cfg/php/enduser/std_alp.php) [30th April 2003]

[27] RHODES N, MCKEEHAN J (1998): **Palm Programming the Developers Guide**. Orielly

[28] Pressman, R.S. (1997). Software Engineering, a practitioner's approach. U.S.A., McGraw Hill.

## Appendix A: The Reflection

### **The Experience**

The project was originally thought up by myself as I had worked with people on my Industrial Placement who had been developing for wireless devices. I found the whole concept fascinating and therefore decided to investigate it more for my final year project.

Starting off I had to find a suitable system that I could improve and in the process investigate wireless technologies. This however proved difficult and I eventually came up with the idea of using vehicle recovery companies. As soon as I started to research the systems I realised that it would be difficult to find information on the companies processes. It was very difficult to locate someone who had the knowledge I required, so I turned to a friend of mine whose father worked for the AA. From then on gathering information became easier.

I have had many problems through this project from the start to finish it has been quite difficult. Firstly I chose to do my own project, which at the time I thought would be good. Looking back however it would have been better to take one of the school projects as with those more help and guidance is available. I found it very difficult adapting to a completely new environment where all processing was based on events. I underestimated the difficulty of the tasks that I originally set myself and underestimated the time it would take to learn the how to develop for mobile devices.

### **What would I do differently?**

If I had the option to do it all over again I would certainly have not chosen my own project without first establishing some methods of help and guidance. I would have more thoroughly investigated what was required to produce the systems that I originally conceived. It would have been helpful to look at previous code speak to some developers etc.

In terms of methodology I would have used a more RAD approach so that I could actually have built more of a system instead of being weighed under with documentation such as SSADM seems to have done. I would have also started much earlier, although I started before Christmas I think it would have been better to have more written up at that point so there is less to do at the end.

## **Conclusions**

In conclusion I have learned many valuable lesson from this project about research, time management and investigating choices more thoroughly. I have enjoyed the work at times and also at other times found it very difficult. I have however learned a lot from the research carried out in the project and in the development. I now believe I have the knowledge to produce systems on Palm devices and may consider developing for other mobile computers.

## Appendix B: Milestones

- 12-12-2002 – Hand in Mid-Project Report: Schedule and initial background reading complete
- 22-12-2002 – Complete background research and chapter
- 22-12-2002 – Complete analysis
- 10-02-2003 – Complete design
- 25-02-2003 – Complete implementation and unit testing
- 10-03-2003 – Complete integration and systems testing
- 21-03-2003 – Progress Meeting
- 31-03-2003 – Complete additional functionality
- 14-04-2003 – Complete Evaluation
- 02-05-2003 – Complete project report and submit

### **Revised Milestones**

- 12-12-2002 – Hand in Mid-Project Report: Schedule and initial background reading complete
- 22-12-2002 – Complete background research and chapter
- 22-12-2002 – Complete analysis
- 03-02-2003 – Complete design
- 21-03-2003 – Progress Meeting
- 02-04-2003 – Complete implementation and unit testing
- 07-04-2003 – Complete integration and systems testing
- 14-04-2003 – Complete additional functionality
- 25-04-2003 – Complete Evaluation
- 02-05-2003 – Complete project report and submit

## Appendix C: Project Schedule

Milestone, Objective	Date Due (DD/MM/YY)
<b>Requirements Analysis and Definition</b>	
Research the existing manual systems used by breakdown companies.	12/12/02
Research other systems that have been adopted to solve similar problems.	12/12/02
Research the wireless hardware options available and decide on the most suitable platform, including the most appropriate OS.	20/12/02
Research the possible development languages for chosen hardware and select most appropriate.	20/12/02
Research the possible database systems that can be used select most appropriate.	22/12/02
Complete the requirements analysis and definition	22/12/02
<b>System and Software Design</b>	
Design of code structure for front end communications module	30/12/02
Design of user interface for front end communications module	03/02/03
Design of database structure	10/02/03
Design of data transmission method	10/02/03
Implementation and Unit Testing	
Coding of front end communications module	17/02/03
Unit testing of front end communications module	19/02/03
Coding of database	24/02/03
Unit testing of database	25/02/03
<b>Integration and Systems Testing</b>	
Coding of data transmission method and connection of database and front end communications module.	5/03/03
Systems Testing	10/03/03
<b>Other</b>	
Extend front end functionality	12/03/03
Store the database on a server and transfer the information from the current desktop to the server via the internet.	17/03/03
Transfer the data using 'wireless' technology directly onto the desktop instead of via a physical cable.	31/03/03
Write up of Report	14/04/03
Write up of user manual	21/04/03

## Revised Project Schedule

Milestone, Objective	Start/Due Date (DD/MM/YY)
<b>Requirements Analysis and Definition</b>	
Research the existing manual systems used by breakdown companies.	12/12/02
Research other systems that have been adopted to solve similar problems.	12/12/02
Research the wireless hardware options available and decide on the most suitable platform, including the most appropriate OS.	20/12/02
Research the possible development languages for chosen hardware and select most appropriate.	20/12/02
Research the possible database systems that can be used select most appropriate.	22/12/02
Complete the requirements analysis and definition	22/12/02
<b>System and Software Design</b>	
Design of code structure for front end communications module	30/12/02
Design of user interface for front end communications module	03/02/03
<b>Implementation and Unit Testing</b>	
Coding of front end communications module	20/02/03 – 31/03/03
Unit testing of front end communications module	31/03/03 – 02/04/03
<b>Integration and Systems Testing</b>	
Coding of data transmission method and connection of database and front end communications module.	02/03/03 –
Systems Testing	07/04/03
<b>Other</b>	
Extend front end functionality	07/04/03 – 11/04/03
Store the information in a database on the desktop and allow data manipulation.	11/04/03 – 14/04/03
Write up of Report	07/04/03 – 25/04/03
Write up of user manual	14/04/03 – 25/04/03



## Appendix D: Interview with AA Patrol Man, Steve Willingham

A phone interview was held with Steve Willingham on the 14<sup>th</sup> November 2002. Mr Willingham is an AA patrolman working in the Blackpool area and has been involved with piloting new systems to improve the AA's current processes and services. The questions asked during the interview were based on the forms AA patrols have to fill in. Copies of these forms were obtained prior to the interview from Mr Willingham and are in Appendix E. The forms consist of a Breakdown Report, two membership application forms and an invoice for parts/service.

### **Question 1:** Why are the current paper forms used?

Mr Willingham stated that there are three main reasons for the Breakdown Report. Firstly it is a means of advertising other products the AA offers to the member. This is done through the advert printed on the back of the copy given to the member. Secondly if the vehicle has to be taken to a garage the Breakdown Report provides the garage with the details of the breakdown and what minor repairs have been done or are needed. Finally it is used as insurance for patrolmen so that they have a record of what they did in case a complaint is made.

The membership forms are used for people who wish to apply for membership. The invoice for parts/service is used as a record of the parts used if any, during the repair. These parts are charged to the customer at an additional rate. The invoice is also the members' receipt for the guarantee of the parts used.

### **Question 2:** How often are the current paper forms used?

According to Mr Willingham "one Breakdown Report is used per breakdown" and the other forms are used as and when needed.

### **Question 3:** What problems have you found using the existing methods and forms?

Mr Willingham did not like having to complete a form at every breakdown. He found that the current Breakdown Report was "not clear, not laid out well and of poor colour", for these reasons many of his colleagues had made complaints. The advert on the customer copy of the report also caused the handwriting to appear incorrectly at times on the second and third copies.

The other problem Mr Willingham found was that one copy of each form had to stay with the patrol for between three to six months this created a large amount of paper that had to be stored.

**Question 4:** Apart from having multiple copies, what do you do with the documents after they have been filled in?

On the copies of the forms obtained prior to the interview the destination of each of the multiple copies was listed. These are explained in section 4.3.2.

The copies held by the patrol are “kept for three to six months and then thrown away” explained Mr Willingham. He then said that they were kept by the patrol in their patrol vehicle, or as he does, in his house. After six months they were disposed of by a specialist service employed by the AA that shredded the forms. This was so that member information was kept confidential.

**Question 5:** What happens to the forms once they are given to the Membership Department, Accounts Department or Suppliers?

Mr Willingham said that the AA has three separate systems that hold information collected by the patrols. These are the Deployment, Breakdown and Membership systems. Currently there is no central system that holds all the data. Therefore the membership application forms have to be sent to the membership department who input the details into the membership system. The invoice for parts/service form has to be sent to the accounts department who enter the details into the accounts system. The details captured in the Breakdown Report are currently not stored on any system.

**Question 6:** If you could improve the current processes what would you change?

Mr Willingham is currently using a laptop used for engine analysis. This is explained further in question 7. He would like to integrate all the processes so that they can be done via the laptop. This would involve having all the forms on the laptop and having wireless connection to the AA's back office systems. The laptop would have the ability to order spares automatically when they are used and automatically send any information captured directly to the AA's systems. All the information held on the member would also be available via the laptop so that details on the

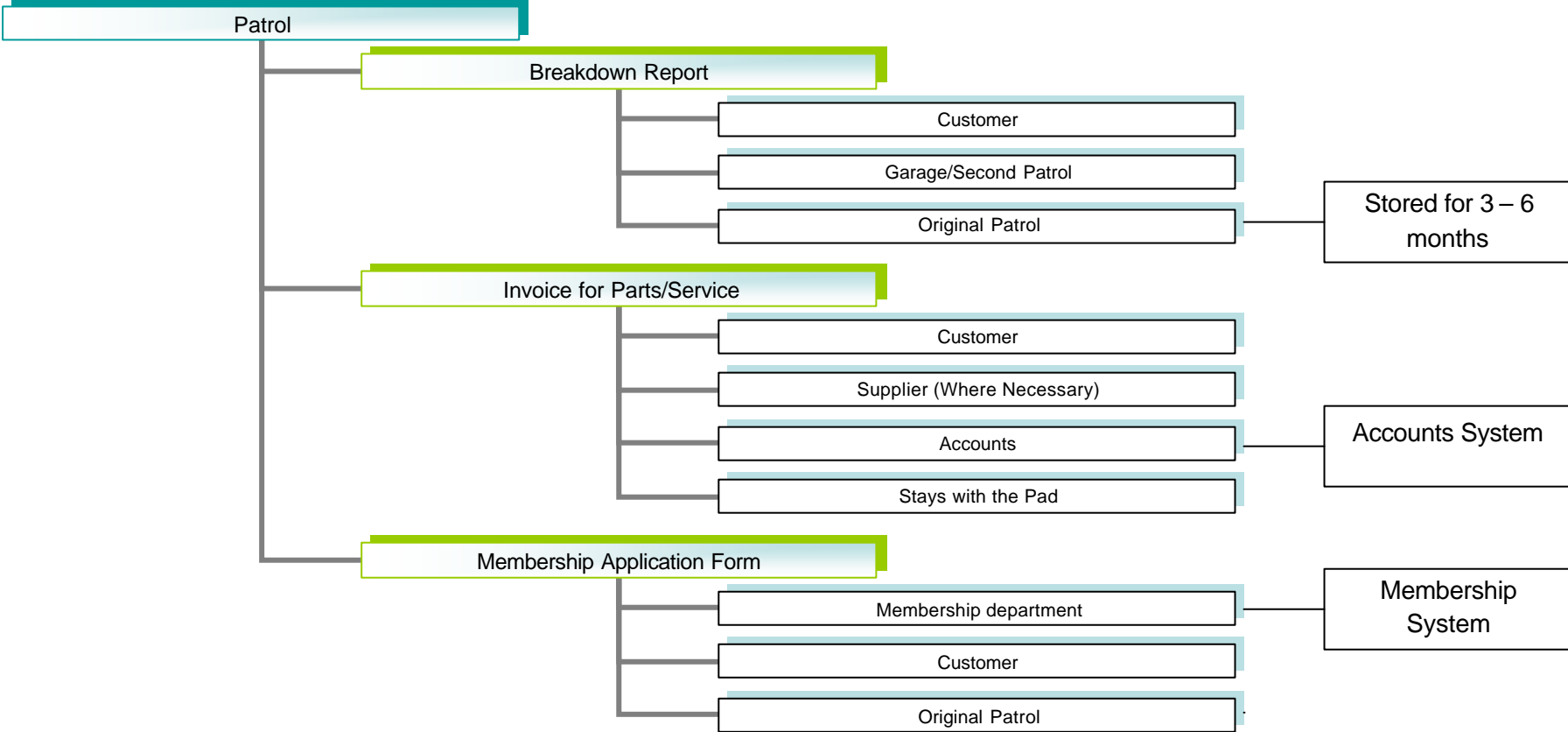
member do not have to be input each time they breakdown. The laptop would also have access to stock lists of the local suppliers (PartCo is the supplier to the AA) so that if a part was required he could check the local supplier's inventory. Each patrol would also have a printer so that they could print off copies of the forms for the customers.

**Question 7:** Are there any methods of improving the current processes currently being implemented? What are these?

"I am currently on the pilot scheme using a laptop for engine analysis" explained Mr Willingham. The laptop is at the moment used for engine analysis only however the functionality may be extended depending on the outcome of the initial trial. This may include transferring the route guidance system used in the vehicles to the laptop instead of having one more than one screen in the cab. The AA is currently in a process of updating and integrating all there systems. They are piloting many schemes to improve their business processes. The Breakdown report is one method that has been introduced to improve the service and record breakdown details.

Mr Willingham was unsure whether the usage of the laptop would become company wide as the actual laptops were of a specialist 'rugged' type costing approximately £3,500. He questioned whether the AA would be willing to purchase such a laptop for each patrol.

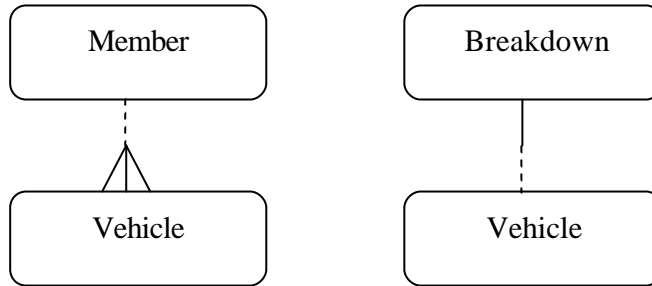
Appendix E: Forms used by AA Patrols and the systems these are entered into



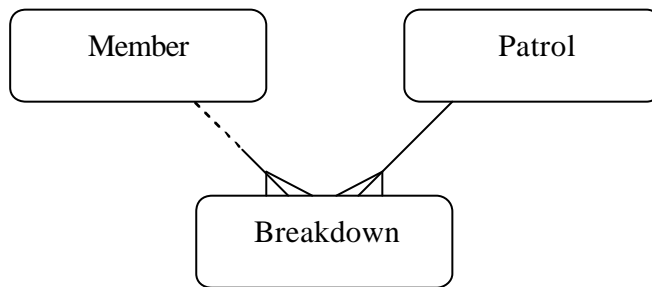


## Appendix F: Logical Data Structures, Data Flow Diagrams

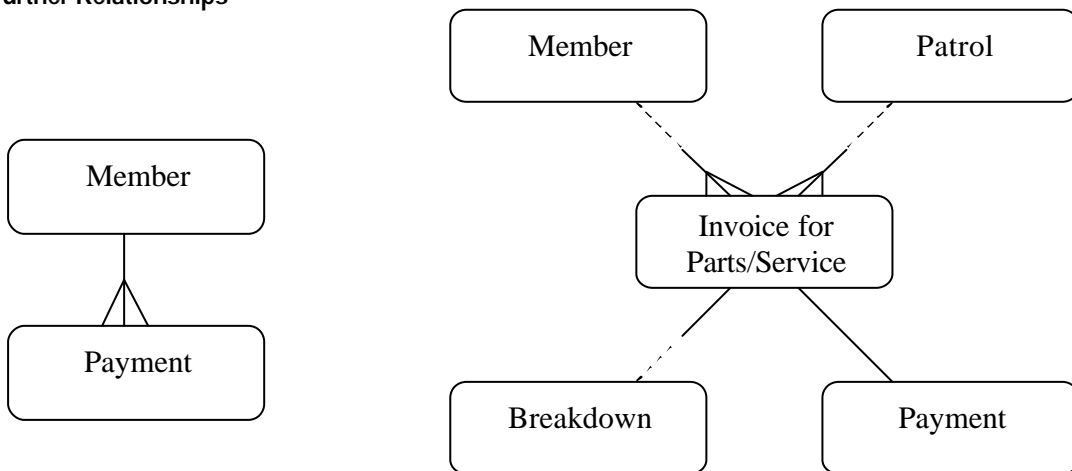
### Essential Relationships



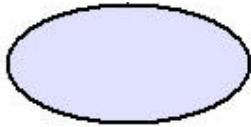
### Desirable Relationships



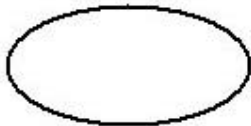
### Further Relationships



**Legend:**



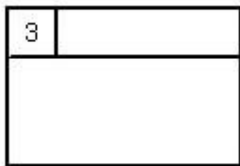
Required Entity



Optional Entity



Required Action



Optional Action

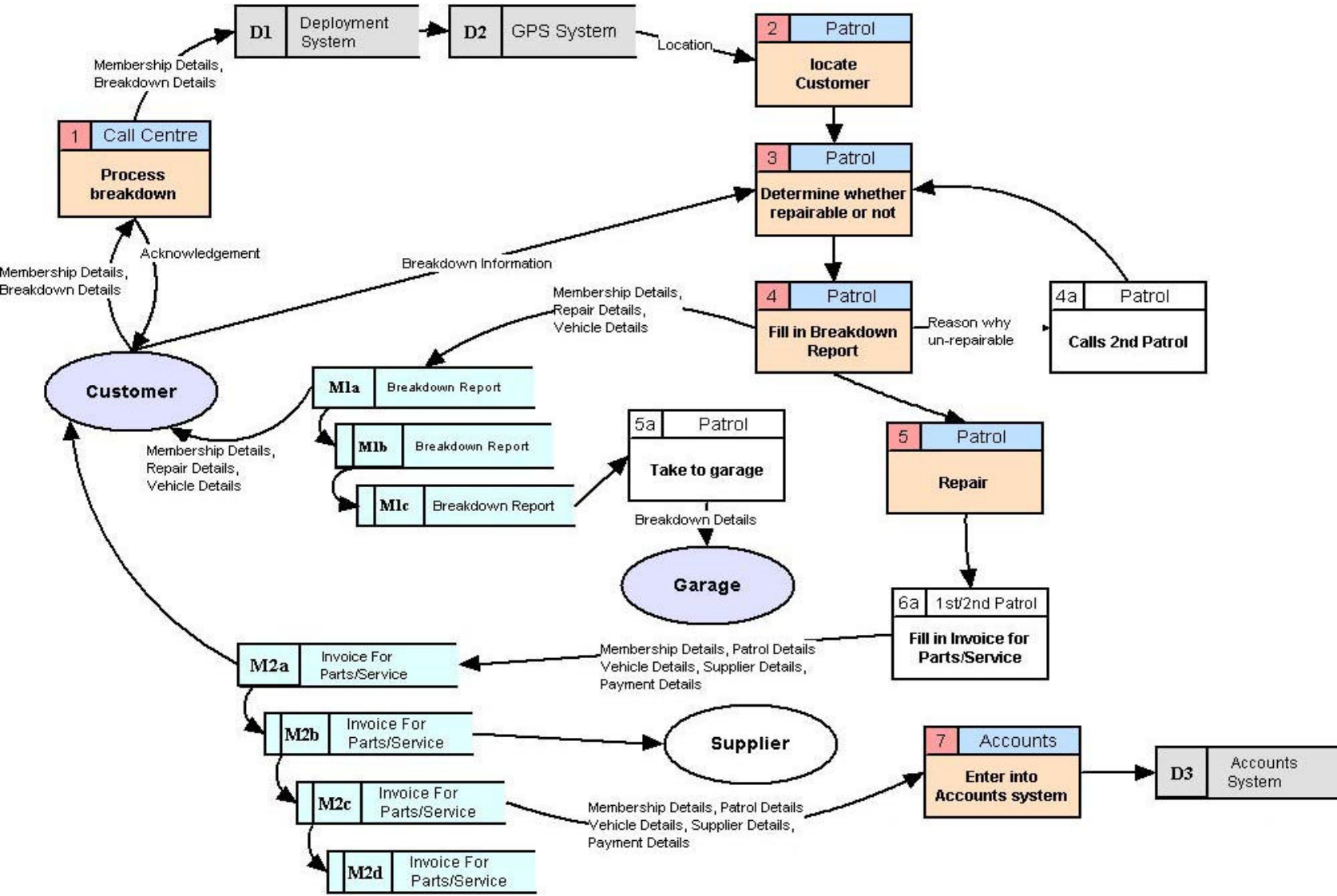


Computerised Data store



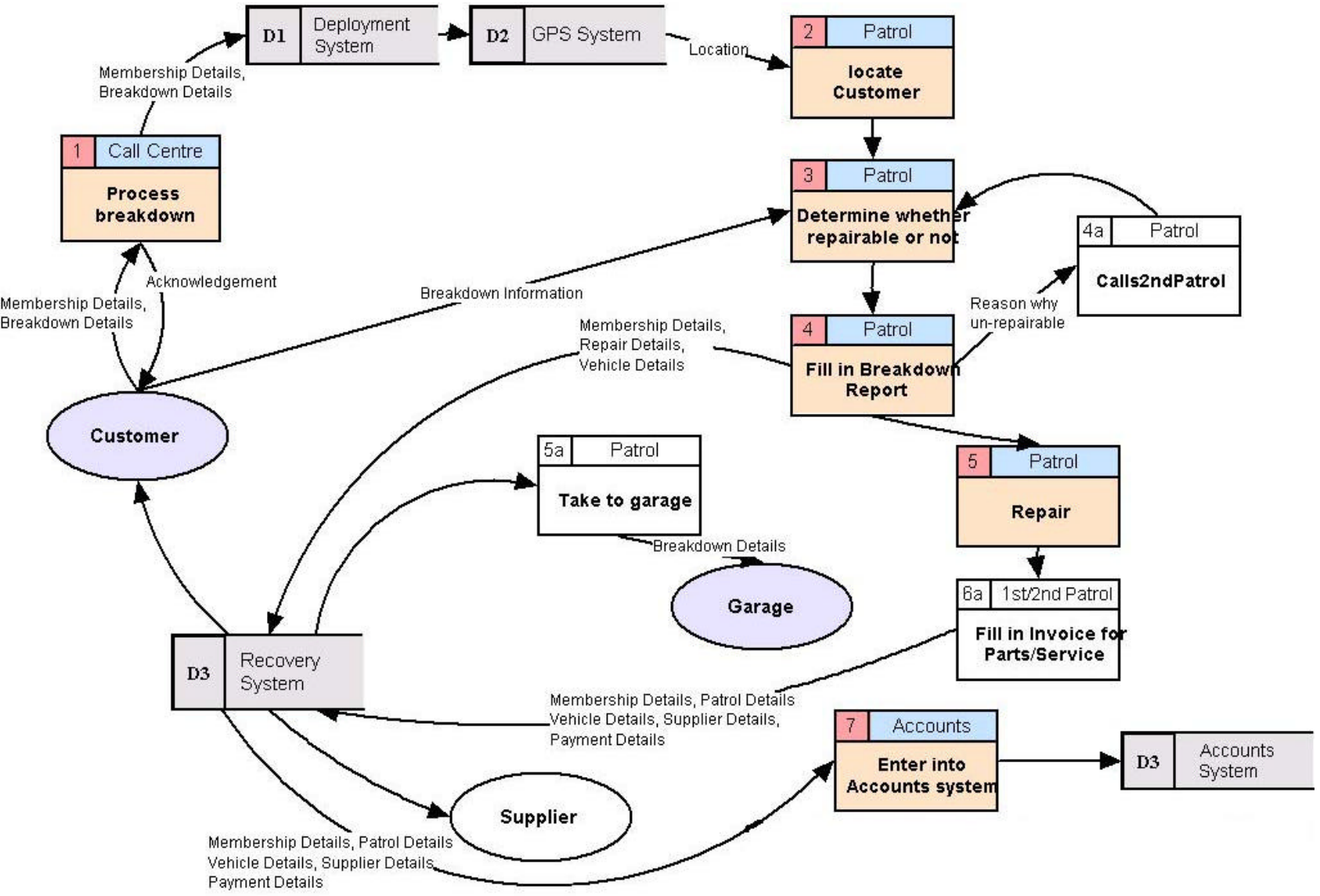
Manual Data Store

# Data Flow Diagram: AA's Existing System





Data Flow Diagram: AA's system after computerisation



## Appendix G: Device Comparison

	Smart Phone	Tablet Computer	Laptop	Handheld
<i>Description</i>	The latest mobile devices to enter into the market. A combination of a mobile phone and a PDA, they offer the functionality and operating system of a PDA whilst allowing wireless access through the mobile telephone networks	Devices that offer some sort of touch screen functionality either through your finger or a stylus. The term tablet computer groups a number of classifications which include webpads and tablet PC's	A laptop computer is simply a portable PC. They all have keyboards, a screen and are clamshell devices.	A handheld computer is as the title suggests a computer that fits in the palm of your hand, they are also known as personal digital assistants or PDA's
<b>Uses</b>	Mobile Phone/PDA	<i>Webpads:</i> for surfing the internet <i>Tablet PC's:</i> Replace traditional laptop PC	Used the same as a normal desktop PC. From word processing to development.	Stores personal information, allows use of 'cut down' PC applications.
<b>User Interface Method / Data entry methods</b>	<ul style="list-style-type: none"> <li>• Stylus pen</li> <li>• Touch screen</li> <li>• Keypad</li> </ul>	<ul style="list-style-type: none"> <li>• Stylus pen</li> <li>• Touch screen</li> <li>• Keyboard</li> </ul>	<ul style="list-style-type: none"> <li>• Keyboard</li> </ul>	<ul style="list-style-type: none"> <li>• Stylus pen</li> <li>• Touch screen</li> <li>• Keyboards (clamshell devices)</li> </ul>
<b>Operating Systems (see section)</b>	<ul style="list-style-type: none"> <li>• Microsoft Windows CE based (Smartphone2002)</li> <li>• Symbian OS</li> </ul>	<ul style="list-style-type: none"> <li>• Microsoft Windows CE based</li> <li>• Microsoft Windows XP Tablet Edition</li> </ul>	<ul style="list-style-type: none"> <li>• Linux based</li> <li>• Microsoft Windows based</li> </ul>	<ul style="list-style-type: none"> <li>• Microsoft Windows CE based (PocketPC2002)</li> <li>• Symbian OS</li> </ul>

	<ul style="list-style-type: none"> <li>• Palm OS</li> </ul>			<ul style="list-style-type: none"> <li>• Palm OS</li> </ul>
<b>Processor Clock Speeds</b>	Between 30 – 130 MHz	Up to 1GHz	Up to 2GHz	Between 16-400MHz
<b>Storage / Memory</b>	Between 2MB –16MB RAM	Up to 255MB RAM, 60GB HD	Up to 512MB RAM, 80GB HD	Between 2MB –16MB RAM
<b>Battery Life</b>	Between 10 hours to 15 days	Between 4 to 12 Hours	Between 3 to 8 hours	Varies considerably: between 6 hours to 2 months
<b>Size and Weight</b>	Normally slightly larger than a normal mobile phone yet smaller than a PDA. Approximately up to 200g	Slightly larger than an A4 piece of paper. Approximately between 1-2Kg	Vary between screen sizes. Tend to be larger than other mobile devices. Approximately between 2-5Kg	Quite small, normally smaller than a hand. Approximately between 100-250g
<b>Wireless Accessibility</b>	<ul style="list-style-type: none"> <li>• GPRS</li> <li>• GSM</li> <li>• Bluetooth</li> </ul>	<ul style="list-style-type: none"> <li>• IEEE 802.11b</li> <li>• Bluetooth</li> </ul>	<ul style="list-style-type: none"> <li>• IEEE 802.11b</li> <li>• Bluetooth</li> </ul>	<ul style="list-style-type: none"> <li>• GPRS</li> <li>• GSM</li> <li>• Bluetooth</li> <li>• IEEE 802.11b</li> </ul>
<b>Development Support</b>	As devices are new developer support and literature limited.	As devices are new developer support and literature is extremely limited.	Developer support is strong as development is the same as a desktop PC.	Developer support is much stronger than smart phones or tablet computers as handheld devices have been produced for some time.





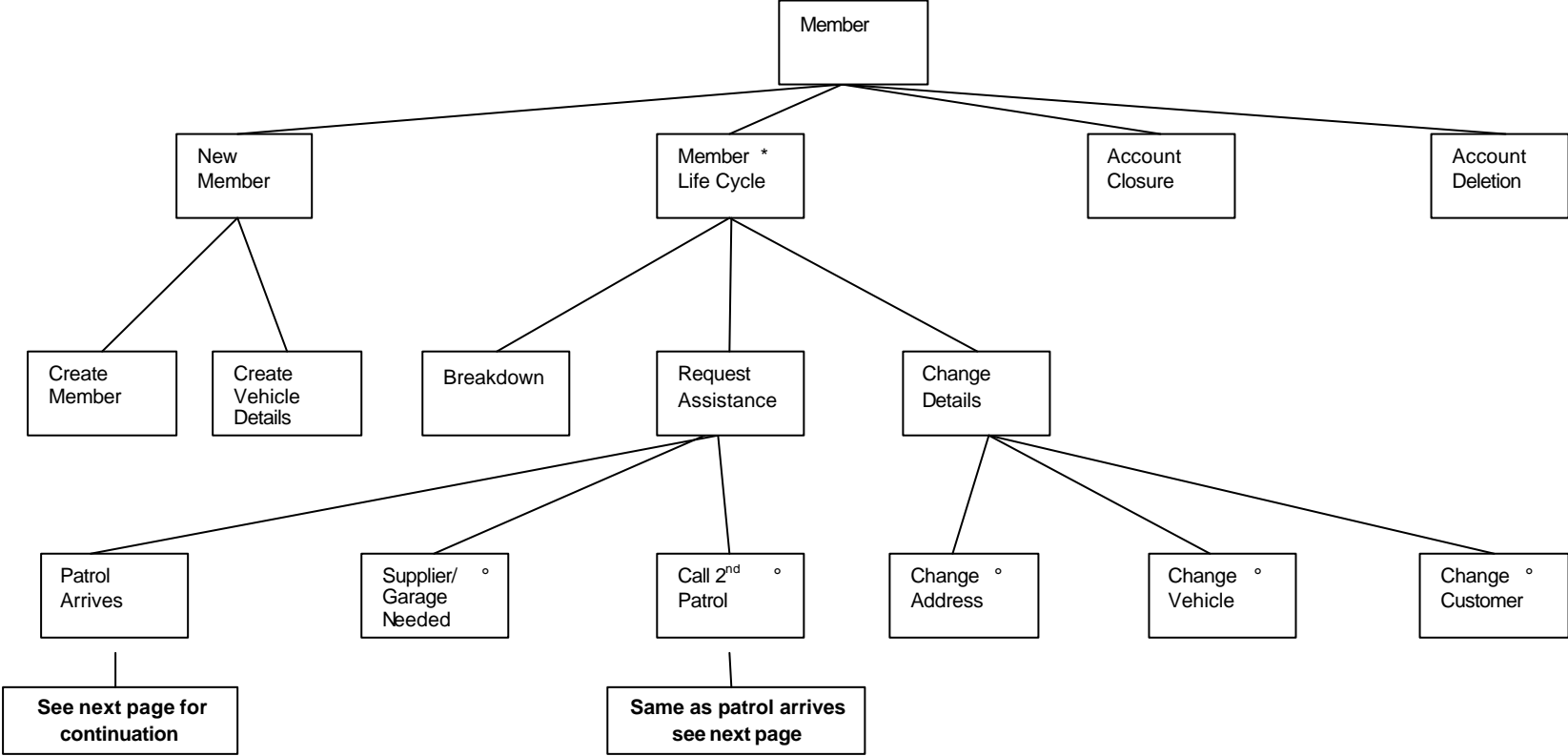
<p><b>Example</b></p>	<p><i>Sony Ericsson P800:</i></p> <ul style="list-style-type: none"> <li>• Symbian OS v7.0</li> <li>• Tri-band GSM and GPRS</li> <li>• Stylus user interface</li> <li>• Infrared and Bluetooth connectivity</li> <li>• Open (J2ME MIDP, PersonalJava and C++)</li> </ul> <p>For more information refer to [15].</p> 	<p><i>Electrovaya Scribbler SC-800:</i></p> <ul style="list-style-type: none"> <li>• Windows XP Tablet</li> <li>• IEEE802.11b WLAN card</li> <li>• 866 MHz Processor</li> <li>• 512MB RAM</li> <li>• 30GB HD</li> <li>• 1.72Kg in weight</li> <li>• 10-16 hours battery life</li> </ul> <p>For more information refer to [16].</p> 	<p><i>Compaq Evo 800c:</i></p> <ul style="list-style-type: none"> <li>• Mobile Intel Pentium 4 Processor-M 1.8GHz</li> <li>• 512MB DDR SDRAM</li> <li>• 30GB or 60GB HD</li> <li>• 2.5kg in weight</li> <li>• Optional IEEE 802.11b or Bluetooth connectivity</li> <li>• Windows XP Pro</li> </ul> <p>For more information refer to [17].</p> 	<p><i>Symbol SPT1800:</i></p> <ul style="list-style-type: none"> <li>• Palm OS 4</li> <li>• IEEE802.11b and GPRS</li> <li>• 16 MB RAM</li> <li>• 33MHz Processor</li> <li>• Full workday capacity battery life.</li> </ul> <p>For more information refer to [12].</p> 
-----------------------	--	--	--	--

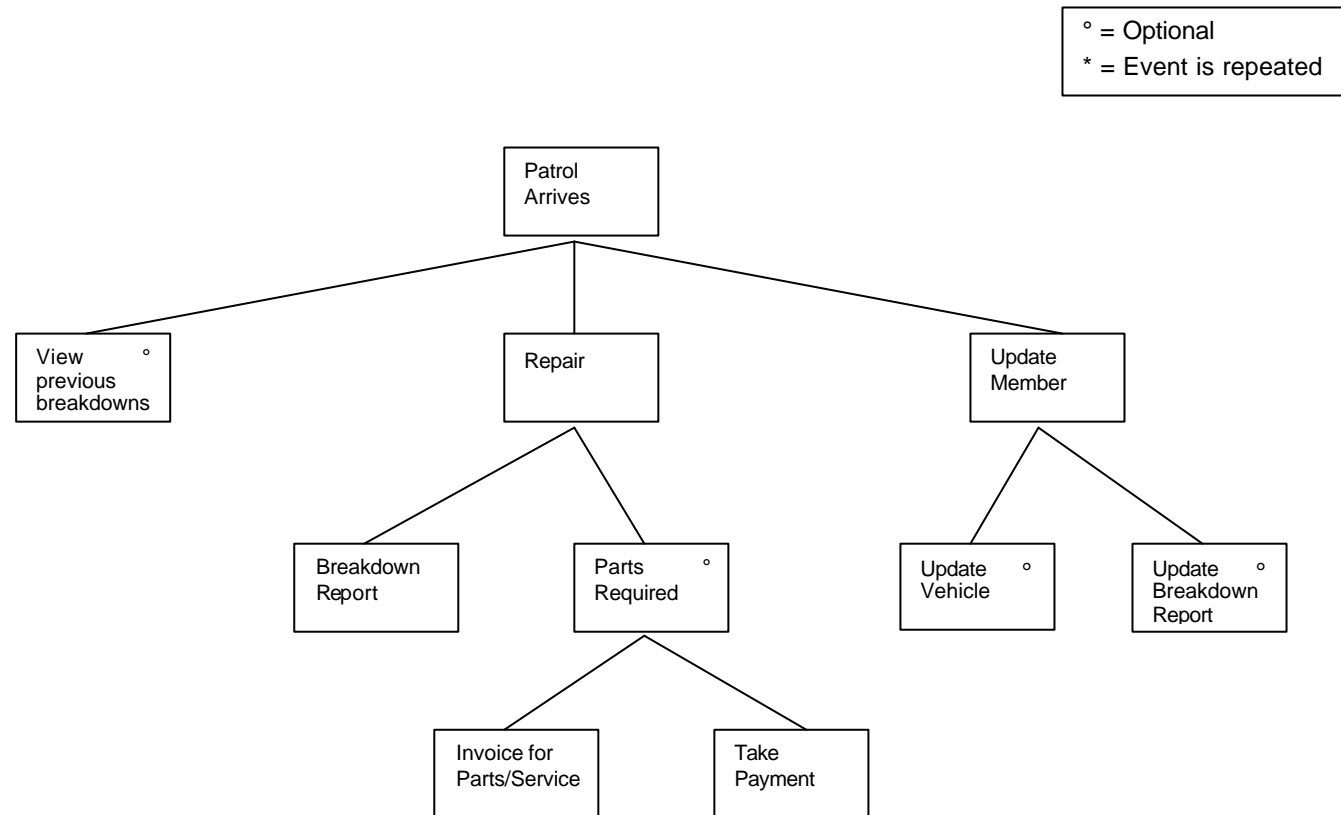
Table 1: Hardware comparison of mobile computers

**Entity Life History Diagram: Vehicle recovery system**

° = Optional  
 \* = Event is



## Entity Life History Diagram: Vehicle recovery system (continued)



# Entity Event Matrix

ENTITY	HEAD OFFICE	PATROL	CUSTOMER
Register new member	X	X	
Change members details	X		X
Change members vehicle details	X	X	X
Close members account	X		X
Delete members account	X		
View previous breakdown details	X	X	
Fill in breakdown report		X	
Fill in Invoice for parts and Service		X	

## Data changes

### Original Data Structure:

#### Member:

- Membership Number: Uint16
- Last Name: Char [20]
- First Name: Char [20]
- DOB: DateType
- Post Code: Char [20]
- Second form of ID: Char[20]

#### Breakdown:

- Fault Type: Char [20] List (Battery, Charging, Electrical, Ignition, Fuel, Cooling, Engine, Exhaust, Transmission, Wheels/Tyres, Lock/Alarms/Immobilisers, Body/Heating/Ventilation, Steering/Suspension/Brakes)
- Other: Char[20]
- Action Taken: Char [20] List (Replaced, Repaired, No Action, Temp Repair)
- Note: Char [255]
- Visited/Not Visited: Boolean
- Job Number: Uint16
- Staff Number: Uint16

#### Vehicle:

- Make: Char [20] List
- Model: Char [20] List
- Mileage: Uint16
- Reg: Char [20]
- Condition Note: Char [255]
- Membership Number: Uint16



## Data Structure Revision 1:

### Member:

- Membership Number: Uint16
- Last Name: Char\*
- First Name: Char\*
- DOB: DateType
- Post Code: Char\*
- Second form of ID: Char\*

### Breakdown:

- Fault Type: Char\* List (Battery, Charging, Electrical, Ignition, Fuel, Cooling, Engine, Exhaust, Transmission, Wheels/Tyres, Security, Body/Vent, Steering, Suspension, Brakes)
- Other: Char\*
- Action Taken: Char\* List (Replaced, Repaired, No Action, Temp Repair)
- Note: Char\*
- Job Number: Uint16
- Visited/Not Visited: Boolean
- Staff Number: Uint16

### Vehicle:

- Make: Char\*
- Model: Char\*
- Reg: Char\*
- Condition Note: Char [255]

Membership Number: Uint16

## Data Structure Revision 2:

Member:

- Membership Number: Uint16
- Last Name: Char\*
- First Name: Char\*
- DOB: DateType
- Post Code: Char\*
- Second form of ID: Char\*

Breakdown:

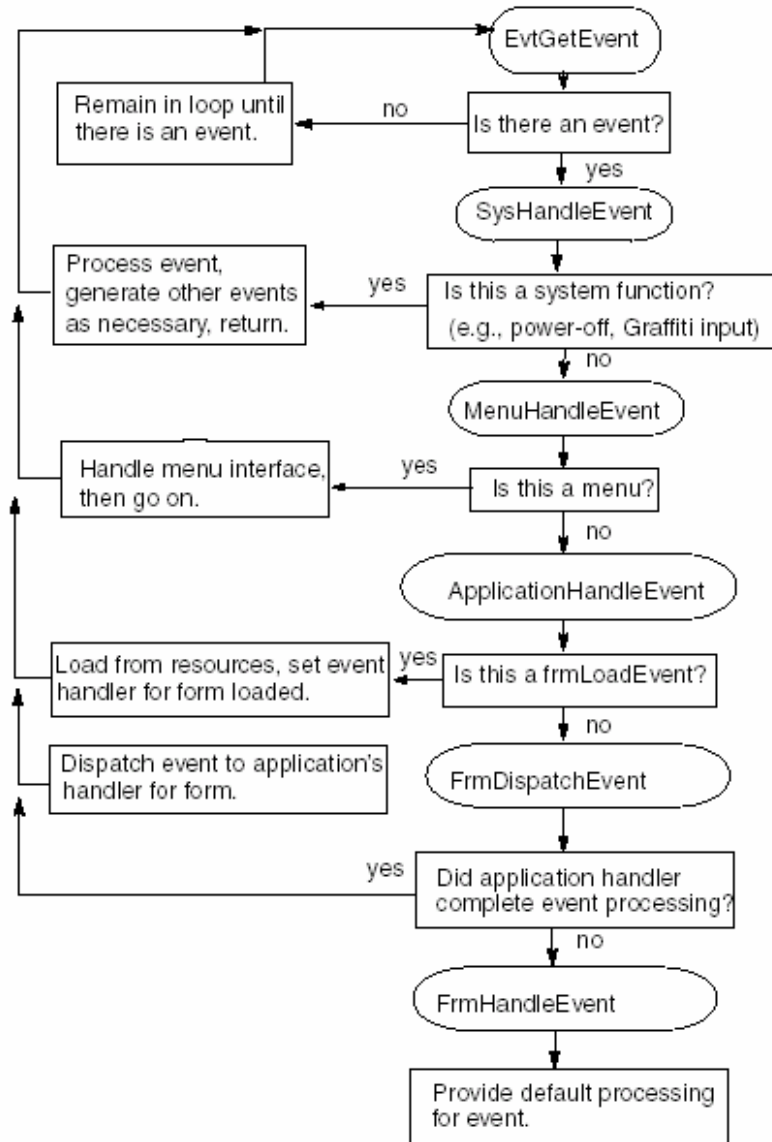
- Fault Type: Char\* List (Battery, Charging, Electrical, Ignition, Fuel, Cooling, Engine, Exhaust, Transmission, Wheels/Tyres, Security, Body/Vent, Steering, Suspension, Brakes)
- Action Taken: Char\* List (Replaced, Repaired, No Action, Temp Repair)
- Visited/Not Visited: Boolean
- Note: Char\*

Vehicle:

- Make: Char\*
- Model: Char\*

Reg: Char\*

## Appendix I: Palm Application Event Loop



## Appendix J: Gremlins Test Results

=====  
This file contains only the last 1024K of generated text.  
=====

```
0.000 (0):
*****
*****
0.000 (0):      ***** Gremlin Hordes started
0.000 (0):
*****
*****
0.000 (0):      Running Gremlins 1 to 15
0.000 (0):      Will run each Gremlin 200 events at a time until all Gremlins have terminated in
error
0.000 (0):      or have reached a maximum of 500 events
0.000 (0):
*****
*****
0.000 (0):      New Gremlin #1 started anew to 200 events
0.911 (201):    Gremlin #1 finished successfully to event #200
0.921: Hordes::PostFakeEvent did not post an event.
0.981: -> EvtAddEventToQueue: appStopEvent
0.991 (0):      New Gremlin #2 started from root state to 200 events
2.223 (203):    Gremlin #2 finished successfully to event #200
2.233: Hordes::PostFakeEvent did not post an event.
2.303: -> EvtAddEventToQueue: appStopEvent
2.303 (0):      New Gremlin #3 started from root state to 200 events
3.124 (203):    Gremlin #3 finished successfully to event #200
3.134: Hordes::PostFakeEvent did not post an event.
3.194: -> EvtAddEventToQueue: appStopEvent
3.194 (0):      New Gremlin #4 started from root state to 200 events
3.965 (201):    Gremlin #4 finished successfully to event #200
3.975: Hordes::PostFakeEvent did not post an event.
4.045: -> EvtAddEventToQueue: appStopEvent
4.045 (0):      New Gremlin #5 started from root state to 200 events
4.746 (201):    Gremlin #5 finished successfully to event #200
4.756: Hordes::PostFakeEvent did not post an event.
4.826: -> EvtAddEventToQueue: appStopEvent
4.826 (0):      New Gremlin #6 started from root state to 200 events
5.718 (202):    Gremlin #6 finished successfully to event #200
5.718 (202):    Hordes::PostFakeEvent did not post an event.
5.788: -> EvtAddEventToQueue: appStopEvent
5.788 (0):      New Gremlin #7 started from root state to 200 events
6.549 (201):    Gremlin #7 finished successfully to event #200
```

6.559: Hordes::PostFakeEvent did not post an event.  
6.619: -> EvtAddEventToQueue: appStopEvent  
6.629 (0): New Gremlin #8 started from root state to 200 events  
7.370 (201): Gremlin #8 finished successfully to event #200  
7.370 (201): Hordes::PostFakeEvent did not post an event.  
7.440: -> EvtAddEventToQueue: appStopEvent  
7.440 (0): New Gremlin #9 started from root state to 200 events  
8.171 (201): Gremlin #9 finished successfully to event #200  
8.181: Hordes::PostFakeEvent did not post an event.  
8.241: -> EvtAddEventToQueue: appStopEvent  
8.251 (0): New Gremlin #10 started from root state to 200 events  
9.053 (202): Gremlin #10 finished successfully to event #200  
9.053 (202): Hordes::PostFakeEvent did not post an event.  
9.123: -> EvtAddEventToQueue: appStopEvent  
9.123 (0): New Gremlin #11 started from root state to 200 events  
9.904 (201): Gremlin #11 finished successfully to event #200  
9.924: Hordes::PostFakeEvent did not post an event.  
9.984: -> EvtAddEventToQueue: appStopEvent  
9.984 (0): New Gremlin #12 started from root state to 200 events  
10.645 (201): Gremlin #12 finished successfully to event #200  
10.655: Hordes::PostFakeEvent did not post an event.  
10.725: -> EvtAddEventToQueue: appStopEvent  
10.725 (0): New Gremlin #13 started from root state to 200 events  
11.286 (202): Gremlin #13 finished successfully to event #200  
11.286 (202): Hordes::PostFakeEvent did not post an event.  
11.356: -> EvtAddEventToQueue: appStopEvent  
11.356 (0): New Gremlin #14 started from root state to 200 events  
11.937 (204): Gremlin #14 finished successfully to event #200  
11.947: Hordes::PostFakeEvent did not post an event.  
12.007: -> EvtAddEventToQueue: appStopEvent  
12.017 (0): New Gremlin #15 started from root state to 200 events  
12.848 (202): Gremlin #15 finished successfully to event #200  
12.868: Hordes::PostFakeEvent did not post an event.  
12.938 (200): Resuming Gremlin #1 to #400 events  
13.669 (401): Gremlin #1 finished successfully to event #400  
13.679: Hordes::PostFakeEvent did not post an event.  
13.739 (200): Resuming Gremlin #2 to #400 events  
14.380 (401): Gremlin #2 finished successfully to event #400  
14.410: Hordes::PostFakeEvent did not post an event.  
14.470 (200): Resuming Gremlin #3 to #400 events  
15.071 (401): Gremlin #3 finished successfully to event #400  
15.081: Hordes::PostFakeEvent did not post an event.  
15.141 (200): Resuming Gremlin #4 to #400 events  
15.842 (401): Gremlin #4 finished successfully to event #400  
15.852: Hordes::PostFakeEvent did not post an event.  
15.912 (200): Resuming Gremlin #5 to #400 events  
16.563 (401): Gremlin #5 finished successfully to event #400

---

16.583: Hordes::PostFakeEvent did not post an event.  
16.654 (200): Resuming Gremlin #6 to #400 events  
17.835 (401): Gremlin #6 finished successfully to event #400  
17.845: Hordes::PostFakeEvent did not post an event.  
17.905 (200): Resuming Gremlin #7 to #400 events  
18.456 (401): Gremlin #7 finished successfully to event #400  
18.456 (401): Hordes::PostFakeEvent did not post an event.  
18.526 (200): Resuming Gremlin #8 to #400 events  
19.027 (401): Gremlin #8 finished successfully to event #400  
19.037: Hordes::PostFakeEvent did not post an event.  
19.097 (200): Resuming Gremlin #9 to #400 events  
20.199 (402): Gremlin #9 finished successfully to event #400  
20.209: Hordes::PostFakeEvent did not post an event.  
20.269 (200): Resuming Gremlin #10 to #400 events  
21.000 (401): Gremlin #10 finished successfully to event #400  
21.010: Hordes::PostFakeEvent did not post an event.  
21.070 (200): Resuming Gremlin #11 to #400 events  
22.342 (401): Gremlin #11 finished successfully to event #400  
22.352: Hordes::PostFakeEvent did not post an event.  
22.422 (200): Resuming Gremlin #12 to #400 events  
23.103 (401): Gremlin #12 finished successfully to event #400  
23.113: Hordes::PostFakeEvent did not post an event.  
23.173 (200): Resuming Gremlin #13 to #400 events  
23.804 (401): Gremlin #13 finished successfully to event #400  
23.814: Hordes::PostFakeEvent did not post an event.  
23.874 (200): Resuming Gremlin #14 to #400 events  
24.425 (401): Gremlin #14 finished successfully to event #400  
24.435: Hordes::PostFakeEvent did not post an event.  
24.505 (200): Resuming Gremlin #15 to #400 events  
25.116 (402): Gremlin #15 finished successfully to event #400  
25.126: Hordes::PostFakeEvent did not post an event.  
25.186 (400): Resuming Gremlin #1 to #500 events  
25.596 (501): Gremlin #1 finished successfully to event #500  
25.596 (501): \*\*\*\*\* Gremlin #1 successfully completed  
25.606: Hordes::PostFakeEvent did not post an event.  
25.666 (400): Resuming Gremlin #2 to #500 events  
26.237 (504): Gremlin #2 finished successfully to event #500  
26.237 (504): \*\*\*\*\* Gremlin #2 successfully completed  
26.247: Hordes::PostFakeEvent did not post an event.  
26.307 (400): Resuming Gremlin #3 to #500 events  
26.828 (502): Gremlin #3 finished successfully to event #500  
26.828 (502): \*\*\*\*\* Gremlin #3 successfully completed  
26.838: Hordes::PostFakeEvent did not post an event.  
26.898 (400): Resuming Gremlin #4 to #500 events  
27.439 (501): Gremlin #4 finished successfully to event #500  
27.439 (501): \*\*\*\*\* Gremlin #4 successfully completed  
27.449: Hordes::PostFakeEvent did not post an event.

---

27.519 (400): Resuming Gremlin #5 to #500 events  
28.000 (501): Gremlin #5 finished successfully to event #500  
28.000 (501): \*\*\*\*\* Gremlin #5 successfully completed  
28.010: Hordes::PostFakeEvent did not post an event.  
28.070 (400): Resuming Gremlin #6 to #500 events  
28.661 (501): Gremlin #6 finished successfully to event #500  
28.661 (501): \*\*\*\*\* Gremlin #6 successfully completed  
28.671: Hordes::PostFakeEvent did not post an event.  
28.731 (400): Resuming Gremlin #7 to #500 events  
29.192 (501): Gremlin #7 finished successfully to event #500  
29.192 (501): \*\*\*\*\* Gremlin #7 successfully completed  
29.202: Hordes::PostFakeEvent did not post an event.  
29.262 (400): Resuming Gremlin #8 to #500 events  
29.732 (501): Gremlin #8 finished successfully to event #500  
29.732 (501): \*\*\*\*\* Gremlin #8 successfully completed  
29.742: Hordes::PostFakeEvent did not post an event.  
29.812 (400): Resuming Gremlin #9 to #500 events  
30.463 (501): Gremlin #9 finished successfully to event #500  
30.463 (501): \*\*\*\*\* Gremlin #9 successfully completed  
30.473: Hordes::PostFakeEvent did not post an event.  
30.533 (400): Resuming Gremlin #10 to #500 events  
30.974 (501): Gremlin #10 finished successfully to event #500  
30.974 (501): \*\*\*\*\* Gremlin #10 successfully completed  
30.984: Hordes::PostFakeEvent did not post an event.  
31.064 (400): Resuming Gremlin #11 to #500 events  
31.585 (501): Gremlin #11 finished successfully to event #500  
31.585 (501): \*\*\*\*\* Gremlin #11 successfully completed  
31.595: Hordes::PostFakeEvent did not post an event.  
31.655 (400): Resuming Gremlin #12 to #500 events  
32.266 (502): Gremlin #12 finished successfully to event #500  
32.266 (502): \*\*\*\*\* Gremlin #12 successfully completed  
32.276: Hordes::PostFakeEvent did not post an event.  
32.336 (400): Resuming Gremlin #13 to #500 events  
32.857 (501): Gremlin #13 finished successfully to event #500  
32.857 (501): \*\*\*\*\* Gremlin #13 successfully completed  
32.867: Hordes::PostFakeEvent did not post an event.  
32.927 (400): Resuming Gremlin #14 to #500 events  
33.297 (501): Gremlin #14 finished successfully to event #500  
33.297 (501): \*\*\*\*\* Gremlin #14 successfully completed  
33.307: Hordes::PostFakeEvent did not post an event.  
33.378 (400): Resuming Gremlin #15 to #500 events  
33.848 (501): Gremlin #15 finished successfully to event #500  
33.848 (501): \*\*\*\*\* Gremlin #15 successfully completed  
33.858 (501): \*\*\*\*\* Gremlin Horde ended at Gremlin #15  
  
33.858 (501): \*\*\*\*\* Device Info:  
33.858 (501): ROM version: 3.1

---

33.858 (501): ROM file name: D:\PalmDev\Emulator\Palm OS 3.1 ROMs\palmos31-en-  
v-dbg.rom  
33.858 (501): Session file: D:\PalmDev\Emulator\library.psf  
33.858 (501): Device name: PalmIIIe  
33.858 (501): RAM size: 2048 KB

33.858 (501): \*\*\*\*\* Error Occurrence Statistics:  
33.858 (501):  
33.858 (501): No Gremlins found errors.



## Appendix K: User Acceptance Testing: 2<sup>nd</sup> Interview with AA Patrol Man, Steve Willingham

A 2<sup>nd</sup> phone interview was held with Steve Willingham on the 13<sup>th</sup> April 2003, so that the user acceptance tests could be carried out and feedback received. Before asking Mr. Willingham the questions prepared, some time was spent explaining the system (as the user manual had not been written yet) and how to use the Palm device. The questions asked during the interview were based on the requirements specifications (see section ????).

**Question 1:** How do you find using the Palm devices user interface?

“It’s a lot simpler than Windows”, replied Mr Willingham. He then went on to say that once he had been told how to operate the device he found that using the stylus to control the menus etc was simpler than using a mouse or keypad especially if you were out on the road. Although he mentioned that the stylus would be very easily misplaced inside a patrol vehicle.

He also said that the device was smaller than he expected.

**Question 2:** What did you think of the Palm devices methods of data entry? Do you prefer them to manual data entry?

After I had explained to Mr Willingham how to access the built in keyboard and use Graffiti he commented on how the entry especially via Graffiti “seems a little confusing”. He found that entry via the built in keyboard was “a little slow” and that using Graffiti would mean, “having to learn English all over again”. He did however find the separate foldable keyboard very easy to use and that it would allow two methods of data entry. He said that he would probably have the keyboard installed into the patrol vehicle and use it their.

Mr Willingham prefers using keyboards than having to manually write out information. He said that he would have to try the other methods more before he could comment further.

**Question 3:** How did you find the appearance of the Recovery software?

“It looks very straight forward”, commented Mr Willingham. He said that the software layout and look was “no frills” and that it was very simple to understand. “It looks like the other applications”, said Mr Willingham. He did comment on the lack of colour and that it would be nice to have a colour screen.

**Question 4:** Did you find the application easy and simple to use?

Mr Willingham found the application was very simple to use and that it was very easy to understand.

**Question 5:** Did Recovery make it easier to generate the breakdown report than using the existing manual system?

Mr Willingham said that the application in its current form made it easier to generate the breakdown form as it was a much simpler than the existing manual system. “Once the info’s in there its there” he commented. He went on to say that only the breakdown details needed to be added as the member details already exist. He did say however that the system lacked some of the functionality of the existing breakdown report and questioned if the system would still be easier if this were to be implemented.

**Question 6:** Do you think that Recovery captured enough information?

Carrying on from the previous question Mr Willingham re-iterated the missing functionality. He said, “It wouldn’t bother me not sure about the AA though”. He also went on to say that it would be nice to have seen the other screens that were supposed to have been implemented. So that he could see what and how a multi function system would work and look like.

**Question 7:** How did you find using the system, was it a steep learning curve? Do you think your colleagues could adapt to such a system easily?

According to Mr Willingham the application would be very “easy to pick up” and that his colleagues could quickly use the system. He did however say that they would not be willing to learn Graffiti and that they would require keyboards or an improved method of data entry.

**Question 8:** Would the Recovery software solve the problem of a 'paper trail'? If the handheld was connected to your other systems would this save you time?

Mr Willingham said that it would solve the problem of a paper trail in terms of the forms that he has to keep at home for six months. He then asked about the multiple copies. I explained to him that they could be printed off using a separate printer and that this functionality could be added. He then commented on the need for the other functionality such as signatures to be implemented, as the copies would have to be authenticated by the member.

In terms of saving time Mr Willingham said that if the application was fully completed and it had the ability to connect to the AA's backend systems then it would stop him having to hand in the copies of the forms. This would ultimately save him time.

**Question 9:** What was your overall impression of the Recovery software? The Palm device?

Mr Willingham thought that the Recovery software was a "bare minimum" application and that it would require a lot more work before it could be seen as a system that could be implemented by the AA. I then asked him if he thought the application would be (after some more additional screens were added) enough for simple data collection and storage such as the breakdown report. He replied by saying "for that I suppose it would be fine".

He thought that the Palm device itself was too limited for the needs of a company such as the AA and it would be more suited to smaller companies or for simple data collections such as the breakdown form.

**Question 10:** Is there anything that you don't like or would like to see added to the software?

Mr Willingham said that it would have been nice to see a few more screens and more functionality. He added that wireless connection to the AA's systems would be ideal so that he would simply have to return to his depot for the information to be transferred.

## Appendix L: Recovery: User Manual

---

**Note:** this user manual does not detail specific Palm OS functions, for those details see the Palm OS user manual.

### 1. Installation

**Note:** To install the system you must have the Palm Desktop Software installed and a cradle linked to the desktop.

To install the system double click on the recovery.prc file and when the installation box appears, check that recovery.prc is listed in the window and click done. The dialogue should indicate that the file will be installed next time you run HotSync.

### 2. Running the software

To open recovery select the following icon:



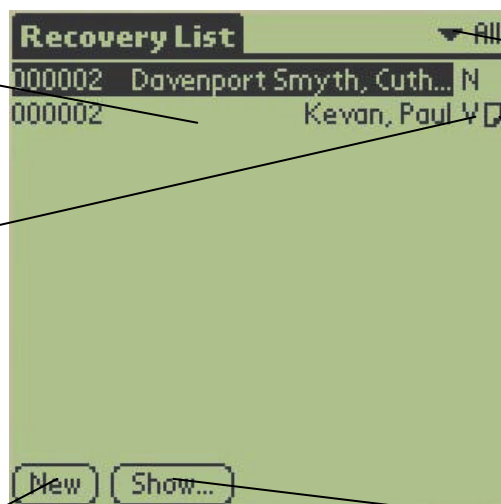
The first screen to appear will be the Recovery List.

### 3. Recovery List Screen

Tap to open details of member

Shows the member has already been visited (V) or not visited (N). Selecting the letter N or V on screen can change this.

Opens the Member Edit screen and creates a new member record



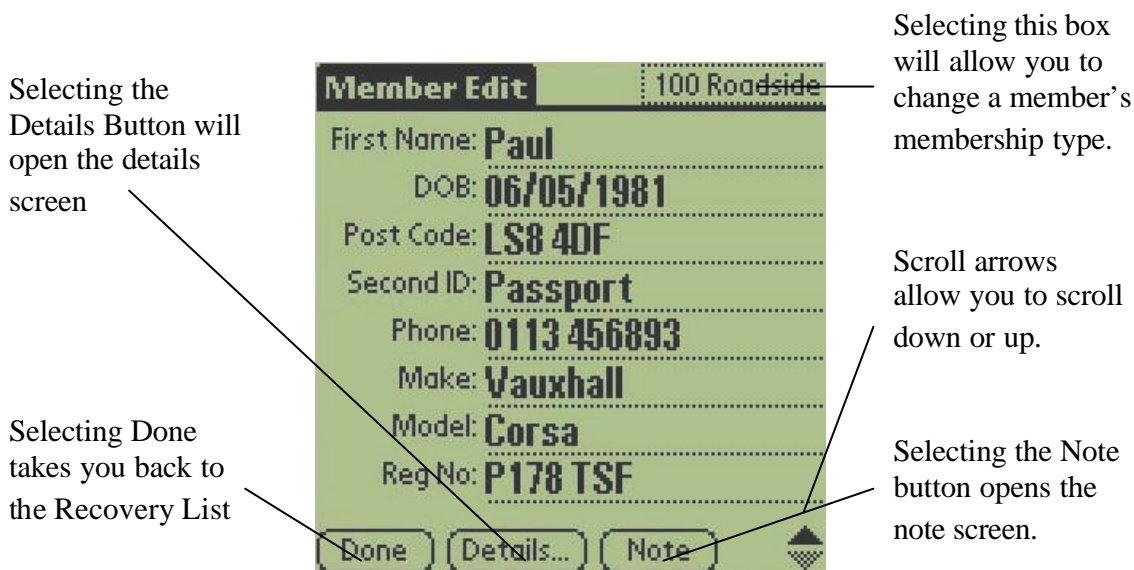
Selecting this arrow will allow you to view the different members by their membership type.

Shows the member has a note attached. Tap on icon to open the note

Opens the application preferences screen

This screen displays the members that are currently held in the system, their details are displayed by membership number, name as default however this can be changed by selecting the Show... button and changing the preferences (see preferences screen). To create a new member record tap the new button and it will open the Member Edit screen with a new record.

#### 4. Member Edit Screen:



The Member Edit screen allows you to add edit and change details of a specific member or enter the details of a new member. The details on the screen are: Membership Number, First Name, Last Name, DOB, Second ID (another form of ID that the member has during breakdown), Phone number, Make, Model and Registration of vehicle.

To enter the breakdown details for the member select the details button and this will display the Breakdown Details screen. To attach a note to the members record, select the Note button and it will display the note screen.

#### 5. Breakdown Details Screen:

The Breakdown details screen allows you to enter basic breakdown details against the member record. These details include the vehicle fault and action taken. The screen also allows the member's membership type to be changed and the member record to be made private. By selecting a record and making it private that record will be hidden when the Palms security is set to hide private records (see your Palm OS user

guide for more details). The visited checkbox is used to keep track of which members have been visited and which have yet to be visited.

Selecting this arrow will allow you to specify the fault of the vehicle

By checking the visited box. The member is tagged as visited.

Selecting the Delete button will delete the current member record.

By Selecting OK you will have saved the breakdown details and the screen will close



Selecting this arrow will allow you to change a member's membership type.

By checking the private box. The record will be made private

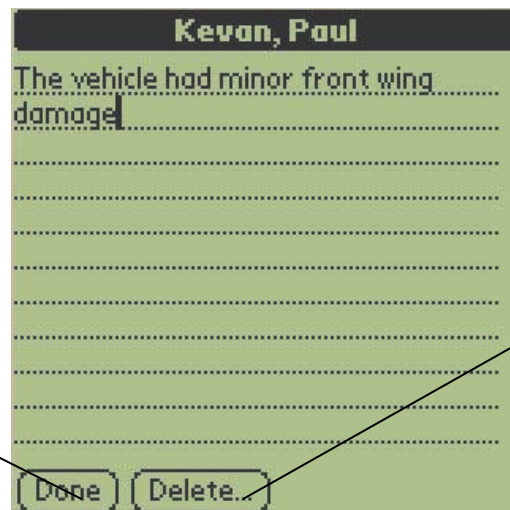
Selecting this arrow will allow you to select the action taken to resolve the fault specified

Selecting the Note button opens the note screen.

The Delete button at the bottom will delete the current member record and the note button will create a note for that record and open the note screen.

## 6. Note Screen

The Done button will take you back to the previous screen.



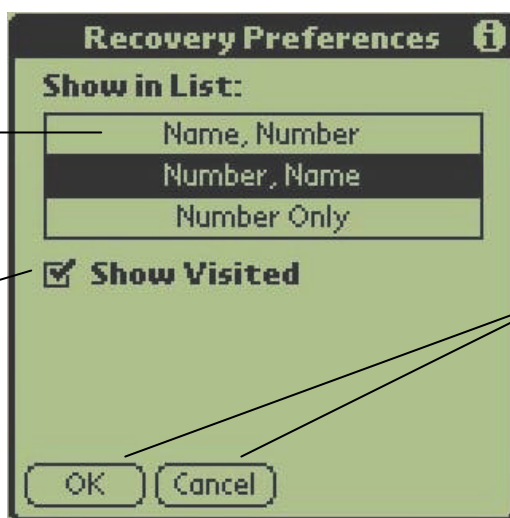
Delete will delete the note from the current member

The Note Screen allows you to add a note to any members record. If nothing is typed then the note will not be saved when the done button is selected.

## 7. Recovery Preferences screen

Selecting one of these changes the layout and sort order of the Recovery List Screen.

By checking the show-visited box. The visited column in the Recovery List screen is viewable.

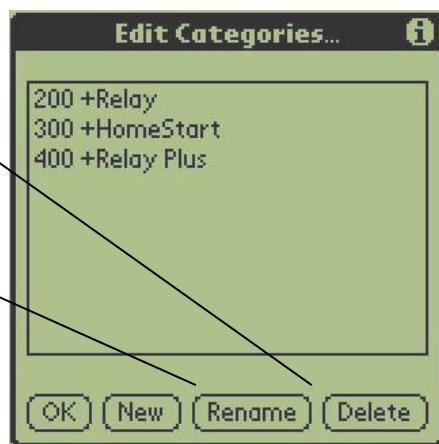


Selecting OK will return you to the Recovery List screen as well as saving the preferences. Cancel will return you to the Recovery List without saving.

The Recovery Preferences screen allows you to select the column order of Recovery List. This also sorts the order of the list to the first column. The show visited check box allows you to specify whether you want to see the visited column in the Recovery List screen.

## 8. Editing the Memberships

Select delete to delete the membership type. Select Rename to rename the membership type



By selecting the membership type in the corner of either the Member Edit or Recovery List screen you can access the Edit Categories screen, which allows you to change the names of the membership types. Consult with your company before you do this. By simply selecting the buttons at the bottom of the edit categories screen you can edit, delete or create a new membership type.

## 9. Menus

Menus are only accessible in the Member Edit Screen and the Note Screen. Both the Member edit screen and the Note screen have general text editing menu options, which includes changing the font size. The Member Edit screen menu also has the functionality to delete a member and attach, detach notes.

