A longitudinal study to formalise a lifecycle methodology for small scale in-house software development

This item was submitted to Loughborough University's Institutional Repository by the/an author.

Additional Information:


Metadata Record: https://dspace.lboro.ac.uk/2134/33093

Publisher: © N.I. Beagley

Rights: This work is made available according to the conditions of the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International (CC BY-NC-ND 4.0) licence. Full details of this licence are available at: https://creativecommons.org/licenses/by-nc-nd/4.0/

Please cite the published version.
A Longitudinal Study to Formalise a Lifecycle Methodology for Small Scale In-House Software Development

by

N I Beagley

A Doctoral Thesis

Submitted in partial fulfilment of the requirements for the award of Doctor of Philosophy of Loughborough University

October 1997

© by N I Beagley 1997
Abstract

Continuing technological advances have brought in-house software development within the reach of small organisations. This study investigates the issues relating to bespoke software development for this emerging user population. It presents a lifecycle methodology, iteratively tailored to the characteristics of small scale in-house development, as a guide to future development.

The study reports the development of the Database Application in Vehicle Ergonomics (DAVE), an information system which was designed to support the specific requirements of a small group of experts. An action research approach was taken through the planning, implementation and evaluation of the project's development methodology. The principles and techniques considered to suit the development scenario were adopted and applied within the framework of a defined lifecycle. The lifecycle directed user centred prototype evolution from a minimal specification to explore and deliver a range of novel applications. Taking a longitudinal approach, the study guided the iterative design of the lifecycle methodology over a series of development and maintenance case studies. The resulting 'modular system' lifecycle successfully managed the limited resources of small scale in-house software development to deliver and support a modular information system that was both usable and used.

The study covers a five year period from project initiation through the activities of specification, development and maintenance. Initial specification of the DAVE system applied the approaches of developer induction, user involvement and historical data analysis to guide the user group's selection of information areas to be supported. Prototype evolution from a minimal specification was used to explore and refine a series of chosen concepts into deliverable products. Formal evaluation based on user testing and expert review provided a safeguard against inappropriate delivery. The project encountered changes in requirements which the lifecycle managed by offering module maintenance as an alternative to continued development. At the end of each cycle the project's continuing work programme was selected by the users based on a system level review.
The project started by identifying the requirements of the user group and the means available to the project with which to support their work. The initial system specification revealed both a group specific requirement and user uncertainty in its definition. The development of test applications provided the project with the strategy of design exploration through evolutionary development in the form of a defined 'prototype development cycle'. A minimal specification of the system's first module, Anthropometry, was refined through the lifecycle's 'module level' specification activities of data set selection, requirement definition and function list construction. From this point, prototype evolution provided an accessible medium for user and developer input to the exploration and refinement of the module's design. Formal evaluation based on user testing and expert evaluation confirmed the prototype's usability whilst highlighting the limitations of background logging and the risk of over development. The lifecycle's system level review led to the development of a 'Standards' module, superseding the group's original priority list. The module's evolution revealed the benefits of expert evaluation as a design activity and direct observation for user testing. The project's third module, Vision, contrasted the 'modular system' lifecycle's approach of design evolution from a minimal specification with the HUFIT toolset's more structured approach to specification. Whilst the toolset provided an appropriate guide for design, prototype evolution was found to make a greater contribution to tailored design within the given scenario. A reduced process based on the lifecycle's development phase was used to upgrade the DAVE system's first module and continue the evolution of the second module to deliver two further text retrieval modules. The assumptions made to reduce the applied effort to a level appropriate to the associated benefits of maintenance are examined. The lifecycle included a floating 'computer support' activity to manage the untimetabled demands of computer use within the small organisation.

The study reviews the success of the DAVE system and its associated lifecycle based on formal evaluation and the objective measure of voluntary usage following delivery. These results are considered against each in-house product's cost in terms of the time allocated to their development. It is concluded that the DAVE development project achieved its goal by providing a series of novel software tools which were accepted and applied by the specified user group to support their work programme requirements. The study presents the characteristics of the development environment and recommends a 'modular system' lifecycle based on user centred prototype evolution from a minimal specification as an appropriate guide for future small scale in-house software development.
Statement

The author was responsible for the planning and execution of all research presented in this thesis, with the exception of the following. Development of the DAVE system's Vision module was carried out as a joint activity between Mr. W. Judd and the author, (chapter 8).
Acknowledgements

This work was made possible by the active involvement and support of the Vehicle Design and Systems group of the Army Personnel Research Establishment, now part of the Centre for Human Sciences, DERA. Valuable guidance was provided by my supervisors, Dr. R. Haslam and Prof. K. Eason. Finally, I owe its completion to the unbounded patience and encouragement of my wife Rosemary.
Table of Contents

Title i
Abstract ii
Statement iv
Acknowledgements v
Table of Contents vi
List of Figures xvii
List of Tables xx
Glossary xxii

1 Introduction 1
   1.1 Overview 1
   1.2 Study Background 2
   1.3 Thesis Aims 2
   1.4 Study Approach 3
   1.5 Thesis Structure 6

2 Development Project Background 8
   2.1 Chapter Overview 8
   2.2 The Army Personnel Research Establishment 8
   2.3 User Group Structure 9
   2.4 The Contract Post 11
   2.5 User Group Work Programme 12
      2.5.1 Applied Ergonomics 12
      2.5.2 Vehicle Related Research 14
   2.6 Computer Usage 15
      2.6.1 Centralised Computing 15
      2.6.2 Personal Computers 16
   2.7 The Centre for Human Sciences 17
3.6 In-House Software Development 43
  3.6.1 User Centred Development 43
  3.6.2 Specification 44
  3.6.3 Design and Coding 46
    3.6.3.1 Prototyping 46
    3.6.3.2 High-Level Software Development 47
  3.6.4 Evaluation 49
  3.6.5 Maintenance 51

3.7 A New Methodology for Small Scale In-House Development 53

4 Specification 54
  4.1 Chapter Overview 54
  4.2 Initial Lifecycle Framework 55
  4.3 Requirements Analysis 57
    4.3.1 Overview 57
    4.3.2 Developer Induction 58
    4.3.3 Informal Discussion 59
    4.3.4 Background Interview 60
    4.3.5 Time Allocation Survey 61
    4.3.6 Time Sheet Analysis 63
    4.3.7 Voluntary Work Log 65
  4.4 Technical Specification 66
    4.4.1 Platform and Tool Assessment 66
    4.4.2 Assessment of Platforms 67
    4.4.3 Assessment of Development Tools 68
    4.4.4 Provisional Platform and Tool Selection 70
  4.5 Product Definition 70
    4.5.1 Formal Group Discussion 70
4.6 Discussion of Methods
4.6.1 User Interview
4.6.2 User Log
4.6.3 Historical Data
4.6.4 Informal Discussion
4.6.5 Developer Induction
4.6.6 Technical Specification
4.6.7 Formal Product Specification
4.6.8 Lifecycle Iteration

4.7 Conclusions

5 Test Application Development
5.1 Chapter Overview
5.2 Acceptance of the Development Tool
5.2.1 Tool Investigation
5.2.2 SuperCard
5.2.3 A Development Process Model
5.3 Test Application Development
5.3.1 Overview
5.3.2 Text Handling
5.3.3 Numerical Data Processing
5.3.4 Interface Support
5.4 The Database Application in Vehicle Ergonomics
5.4.1 System Structure
5.4.2 DAVE Entry Screen
5.5 Discussion of Methods
5.5.1 The Contribution of Test Application Development
5.5.2 Development Experience
5.5.3 Indication of Scope
5.5.4 Product Utility
5.5.5 The DAVE Entry Screen
5.5.6 The Prototype Development Cycle
5.6 Conclusions
# The Anthropometry Module

## Chapter Overview

## Module Development Lifecycle

## Anthropometry Module Specification

### Data Set Selection

### Requirements Definition

### Function List Construction

## Prototype Development Cycle

### The Non-Functional Prototype

### Design of Functional Elements

### Interface Form and Structure

### Completion of the Prototype Development Cycle

## Acceptance

### Expert Evaluation

### Formal User Evaluation

#### Aims

#### Method

#### Results

### Design Iteration

## Delivery

## Discussion of Methods

### Module Specification

### Module Development

### Module Acceptance

#### Expert Evaluation

#### User Evaluation

### Delivery and Maintenance

## Lifecycle Redesign for a Modular System

## Conclusions
### 7.8 Discussion of Methods

- **7.8.1 'System Level' Specification** 171
- **7.8.2 'Module Level' Specification** 174
- **7.8.3 Prototyping** 176
- **7.8.4 Acceptance** 179

### 7.9 Conclusions 180

### 8 The Vision Module 182

- **8.1 Chapter Overview** 182
- **8.2 Lifecycle Approach** 183
  - **8.2.1 Parallel Prototype Development** 183
  - **8.2.2 The HUFIT Toolset** 185
- **8.3 System Specification** 186
- **8.4 Prototype A - Modular Development Lifecycle** 188
  - **8.4.1 Module Specification** 188
  - **8.4.2 Prototype Evolution** 189
- **8.5 Prototype B - HUFIT Toolset** 191
  - **8.5.1 Module Specification** 191
  - **8.5.2 Prototype Design** 192
- **8.6 Comparative Evaluation** 194
  - **8.6.1 Aim** 194
  - **8.6.2 Method** 194
  - **8.6.3 Results** 195
- **8.7 Design Integration** 197
- **8.8 Acceptance** 199
  - **8.8.1 Expert Evaluation** 200
  - **8.8.2 Formal User Evaluation** 201
- **8.9 Delivery and Maintenance** 203
8.10 Discussion of Methods

8.10.1 Programme Selection .......................... 205
8.10.2 Prototype Specification ......................... 206
8.10.3 Prototype Development ......................... 208
8.10.4 Design Integration ............................ 210
8.10.5 Expert Evaluation .................................. 211
8.10.6 Formal User Testing ............................ 212

8.11 Lifecycle Overview .................................. 213

8.12 Conclusions ...................................... 215

9 Maintenance and Support ................................. 217

9.1 Chapter Overview .................................. 217

9.2 The Maintenance Phase ............................. 218

9.3 The Structural Evolution of the DAVE System ...... 219

9.3.1 The Entry Screen ................................ 220
9.3.2 System Redesign ................................ 221
9.3.3 Direct Module Access ............................. 222

9.4 Anthropometry Module Upgrade ...................... 223

9.4.1 Module Re-specification .......................... 223
9.4.2 Module Redesign ................................ 224
9.4.3 Delivery and Support ............................ 226

9.5 Alternative Evolutionary Paths ...................... 226

9.5.1 Expanding the Text Database .................... 226
9.5.2 The General Module .............................. 227
9.5.3 The Questionnaire Module ....................... 229
9.5.4 Staff Targets .................................... 230

9.6 Computer Support .................................. 231

9.6.1 Error Correction ................................ 232
9.6.2 Hardware Maintenance ........................... 233
9.6.3 User Instruction ................................ 234
<table>
<thead>
<tr>
<th>10.6</th>
<th>Discussion of Methods</th>
<th>270</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.6.1</td>
<td>Project Selection and Module Specification</td>
<td>270</td>
</tr>
<tr>
<td>10.6.2</td>
<td>Evolutionary Design and Development</td>
<td>272</td>
</tr>
<tr>
<td>10.6.3</td>
<td>Maintenance and Computer Support</td>
<td>273</td>
</tr>
<tr>
<td>10.7</td>
<td>Conclusions</td>
<td>274</td>
</tr>
</tbody>
</table>

11 A 'Modular System' Lifecycle

<table>
<thead>
<tr>
<th>11.1</th>
<th>Chapter Overview</th>
<th>276</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.2</td>
<td>Background</td>
<td>276</td>
</tr>
<tr>
<td>11.2.1</td>
<td>The Scenario</td>
<td>277</td>
</tr>
<tr>
<td>11.2.2</td>
<td>The Study</td>
<td>277</td>
</tr>
<tr>
<td>11.2.3</td>
<td>The Development Project</td>
<td>279</td>
</tr>
<tr>
<td>11.2.4</td>
<td>User Participation</td>
<td>279</td>
</tr>
<tr>
<td>11.3</td>
<td>A 'Modular System' Lifecycle</td>
<td>281</td>
</tr>
<tr>
<td>11.3.1</td>
<td>Model Design and Evolution</td>
<td>281</td>
</tr>
<tr>
<td>11.3.2</td>
<td>Phases of the 'Modular System' Lifecycle</td>
<td>285</td>
</tr>
<tr>
<td>11.4</td>
<td>The 'System Level' Specification Phase</td>
<td>287</td>
</tr>
<tr>
<td>11.4.1</td>
<td>Background Analysis</td>
<td>288</td>
</tr>
<tr>
<td>11.4.2</td>
<td>Platform and Tool Selection</td>
<td>289</td>
</tr>
<tr>
<td>11.4.3</td>
<td>Project Overview</td>
<td>290</td>
</tr>
<tr>
<td>11.4.4</td>
<td>Work Programme Specification</td>
<td>291</td>
</tr>
<tr>
<td>11.5</td>
<td>The 'Module Level' Specification Phase</td>
<td>292</td>
</tr>
<tr>
<td>11.5.1</td>
<td>Module Specification</td>
<td>292</td>
</tr>
<tr>
<td>11.5.2</td>
<td>Module Re-Specification</td>
<td>294</td>
</tr>
<tr>
<td>11.5.3</td>
<td>HUFIT Tools for Specification</td>
<td>294</td>
</tr>
<tr>
<td>11.6</td>
<td>The Prototyping Phase</td>
<td>296</td>
</tr>
<tr>
<td>11.6.1</td>
<td>The Prototype Development Cycle</td>
<td>296</td>
</tr>
<tr>
<td>11.6.2</td>
<td>User Input to the Development Cycle</td>
<td>298</td>
</tr>
<tr>
<td>11.6.3</td>
<td>Expert Evaluation within the Development Cycle</td>
<td>299</td>
</tr>
<tr>
<td>11.6.4</td>
<td>Object Oriented Programming</td>
<td>300</td>
</tr>
<tr>
<td>11.6.5</td>
<td>Design Issues</td>
<td>301</td>
</tr>
<tr>
<td>11.6.6</td>
<td>The Influence of Individual Characteristics</td>
<td>302</td>
</tr>
</tbody>
</table>
List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Organisational Structure of APRE (1990)</td>
<td>9</td>
</tr>
<tr>
<td>2.2</td>
<td>User Group Hierarchy</td>
<td>10</td>
</tr>
<tr>
<td>2.3</td>
<td>Restricted Crewstation Access in a Vehicle Mock-up</td>
<td>13</td>
</tr>
<tr>
<td>2.4</td>
<td>Prototype Map Fitted to a Tank</td>
<td>15</td>
</tr>
<tr>
<td>2.5</td>
<td>Organisational Structure of the Centre for Human Sciences 1995</td>
<td>19</td>
</tr>
<tr>
<td>3.1</td>
<td>Software Phases</td>
<td>40</td>
</tr>
<tr>
<td>3.2</td>
<td>Waterfall Model</td>
<td>41</td>
</tr>
<tr>
<td>3.3</td>
<td>The Spiral Model of Software Development and Enhancement</td>
<td>42</td>
</tr>
<tr>
<td>3.4</td>
<td>Rapid Prototyping Model</td>
<td>43</td>
</tr>
<tr>
<td>4.1</td>
<td>Development Project Lifecycle Model</td>
<td>56</td>
</tr>
<tr>
<td>4.2</td>
<td>Specification Phase</td>
<td>56</td>
</tr>
<tr>
<td>4.3</td>
<td>Perceived Time Allocation</td>
<td>62</td>
</tr>
<tr>
<td>4.4</td>
<td>Total Time Allocation (1989 to 1990)</td>
<td>64</td>
</tr>
<tr>
<td>4.5</td>
<td>Technical Specification</td>
<td>66</td>
</tr>
<tr>
<td>4.6</td>
<td>Lifecycle Model for Design Exploration</td>
<td>84</td>
</tr>
<tr>
<td>4.7</td>
<td>Provisional Lifecycle Model</td>
<td>84</td>
</tr>
<tr>
<td>5.1</td>
<td>The Structure of a SuperCard Project</td>
<td>88</td>
</tr>
<tr>
<td>5.2</td>
<td>Prototyping Model</td>
<td>90</td>
</tr>
<tr>
<td>5.3</td>
<td>Text Application</td>
<td>93</td>
</tr>
<tr>
<td>5.4</td>
<td>Numerical Test Application</td>
<td>95</td>
</tr>
<tr>
<td>5.5</td>
<td>Interface Application</td>
<td>97</td>
</tr>
<tr>
<td>5.6</td>
<td>DAVE System Structure</td>
<td>99</td>
</tr>
<tr>
<td>5.7</td>
<td>DAVE Entry Screen</td>
<td>100</td>
</tr>
<tr>
<td>5.8</td>
<td>Prototype Development Cycle</td>
<td>105</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>6.1</td>
<td>Evolutionary Development Lifecycle</td>
<td>110</td>
</tr>
<tr>
<td>6.2</td>
<td>Early Anthropometry Prototype</td>
<td>116</td>
</tr>
<tr>
<td>6.3</td>
<td>Photograph ‘Hot-spots’</td>
<td>117</td>
</tr>
<tr>
<td>6.4</td>
<td>Directional ‘Hot-spots’</td>
<td>117</td>
</tr>
<tr>
<td>6.5</td>
<td>Anthropometry Prototype</td>
<td>118</td>
</tr>
<tr>
<td>6.6</td>
<td>Acceptance Phase</td>
<td>120</td>
</tr>
<tr>
<td>6.7</td>
<td>Implemented Anthropometry Module</td>
<td>127</td>
</tr>
<tr>
<td>6.8</td>
<td>Delivery Phase</td>
<td>129</td>
</tr>
<tr>
<td>6.9</td>
<td>Lifecycle for the Evolution of a Modular System</td>
<td>140</td>
</tr>
<tr>
<td>7.1</td>
<td>Framework for the Modular System Lifecycle</td>
<td>145</td>
</tr>
<tr>
<td>7.2</td>
<td>Module Specification Phase</td>
<td>150</td>
</tr>
<tr>
<td>7.3</td>
<td>Standards Prototype's Index Screen</td>
<td>158</td>
</tr>
<tr>
<td>7.4</td>
<td>Standards Prototype's Document Screen</td>
<td>158</td>
</tr>
<tr>
<td>7.5</td>
<td>Implemented Standards Module's Index Screen</td>
<td>169</td>
</tr>
<tr>
<td>7.6</td>
<td>Implemented Standards Module's Document Screen</td>
<td>169</td>
</tr>
<tr>
<td>7.7</td>
<td>System Specification Outline</td>
<td>173</td>
</tr>
<tr>
<td>7.8</td>
<td>Prototyping Phase incorporating the Prototype Development Cycle</td>
<td>178</td>
</tr>
<tr>
<td>8.1</td>
<td>Parallel Prototyping Approach for Vision Module Development</td>
<td>184</td>
</tr>
<tr>
<td>8.2</td>
<td>The HUFIT Planning, Analysis and Specification Toolset (HUSAT 1989)</td>
<td>185</td>
</tr>
<tr>
<td>8.3</td>
<td>Information Input Screen (Prototype A)</td>
<td>190</td>
</tr>
<tr>
<td>8.4</td>
<td>Data Input Screen (Prototype A)</td>
<td>191</td>
</tr>
<tr>
<td>8.5</td>
<td>Information Input Screen (Prototype B)</td>
<td>193</td>
</tr>
<tr>
<td>8.6</td>
<td>Data Input Screen (Prototype B)</td>
<td>193</td>
</tr>
<tr>
<td>8.7</td>
<td>Evaluated Prototype's Task Completion Time and Standard Deviation</td>
<td>195</td>
</tr>
<tr>
<td>8.8</td>
<td>Vision Module Information Input Screen</td>
<td>198</td>
</tr>
<tr>
<td>8.9</td>
<td>Vision Module Data Input Screen</td>
<td>198</td>
</tr>
<tr>
<td>8.10</td>
<td>Vision Module Plot Output Screen</td>
<td>199</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>9.1</td>
<td>Process Model for Product Maintenance</td>
<td>219</td>
</tr>
<tr>
<td>9.2</td>
<td>DAVE Entry Screen Redesign</td>
<td>222</td>
</tr>
<tr>
<td>9.3</td>
<td>Customised icons for the DAVE System Modules</td>
<td>223</td>
</tr>
<tr>
<td>9.4</td>
<td>Anthropometry Module Interface Redesign</td>
<td>225</td>
</tr>
<tr>
<td>9.5</td>
<td>General Module Index Screen</td>
<td>228</td>
</tr>
<tr>
<td>9.6</td>
<td>General Module's Indexing Utility</td>
<td>228</td>
</tr>
<tr>
<td>9.7</td>
<td>DAVE System Evolution to Meet Changing Requirements</td>
<td>240</td>
</tr>
<tr>
<td>9.8</td>
<td>Anthropometry Module Evolution to Meet Changing Requirements</td>
<td>242</td>
</tr>
<tr>
<td>9.9</td>
<td>Maintenance and Support within the 'Modular System' Lifecycle</td>
<td>244</td>
</tr>
<tr>
<td>10.1</td>
<td>DAVE Module Usage</td>
<td>251</td>
</tr>
<tr>
<td>10.2</td>
<td>Total In-House Software Usage</td>
<td>258</td>
</tr>
<tr>
<td>10.3</td>
<td>Developer Time Allocation</td>
<td>266</td>
</tr>
<tr>
<td>11.1</td>
<td>The 'Modular System' Lifecycle Framework</td>
<td>281</td>
</tr>
<tr>
<td>11.2</td>
<td>Phases of the 'Modular System' Lifecycle</td>
<td>285</td>
</tr>
<tr>
<td>11.3</td>
<td>The 'System Level' Specification Phase</td>
<td>287</td>
</tr>
<tr>
<td>11.4</td>
<td>'Module Level' Specification Phase</td>
<td>292</td>
</tr>
<tr>
<td>11.5</td>
<td>The Prototyping Phase</td>
<td>296</td>
</tr>
<tr>
<td>11.6</td>
<td>The Acceptance Phase</td>
<td>303</td>
</tr>
<tr>
<td>11.7</td>
<td>The Delivery Phase</td>
<td>308</td>
</tr>
<tr>
<td>11.8</td>
<td>The 'Modular System' Lifecycle</td>
<td>313</td>
</tr>
</tbody>
</table>
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Informal Techniques for Requirements Definition</td>
<td>57</td>
</tr>
<tr>
<td>4.2</td>
<td>Formal Techniques for Requirements Definition</td>
<td>58</td>
</tr>
<tr>
<td>4.3</td>
<td>Structured Interview Responses.</td>
<td>61</td>
</tr>
<tr>
<td>4.4</td>
<td>Assessment of Computer Platforms</td>
<td>68</td>
</tr>
<tr>
<td>4.5</td>
<td>Prioritised List of VDS Information Areas for Support.</td>
<td>72</td>
</tr>
<tr>
<td>4.6</td>
<td>Sources for Requirements Analysis</td>
<td>73</td>
</tr>
<tr>
<td>5.1</td>
<td>Functional Specification for the Text Application</td>
<td>92</td>
</tr>
<tr>
<td>5.2</td>
<td>Functional Specification for the Numerical Application</td>
<td>94</td>
</tr>
<tr>
<td>6.1</td>
<td>Anthropometry Function List</td>
<td>114</td>
</tr>
<tr>
<td>6.2</td>
<td>Recommendations and Actions following Expert Evaluation</td>
<td>121</td>
</tr>
<tr>
<td>6.3</td>
<td>Question Sheet Results</td>
<td>124</td>
</tr>
<tr>
<td>6.4</td>
<td>User Comments</td>
<td>125</td>
</tr>
<tr>
<td>6.5</td>
<td>Question Completion Time and Action Sequence Ranges</td>
<td>126</td>
</tr>
<tr>
<td>6.6</td>
<td>Design Problems and Actions following Formal User Evaluation</td>
<td>127</td>
</tr>
<tr>
<td>7.1</td>
<td>VDS Reference Documents</td>
<td>151</td>
</tr>
<tr>
<td>7.2</td>
<td>Standards Module Function List</td>
<td>153</td>
</tr>
<tr>
<td>7.3</td>
<td>User Evaluation Question matrix</td>
<td>161</td>
</tr>
<tr>
<td>7.4</td>
<td>Completion Times (s) and Errors for the User Evaluation</td>
<td>162</td>
</tr>
<tr>
<td>7.5</td>
<td>Summary of Completion Times (s) and Errors for the User Evaluation</td>
<td>163</td>
</tr>
<tr>
<td>7.6</td>
<td>Task Actions and Errors for the User Evaluation</td>
<td>164</td>
</tr>
<tr>
<td>7.7</td>
<td>Summary of Task Actions and Errors for the User Evaluation</td>
<td>165</td>
</tr>
<tr>
<td>7.8</td>
<td>The User Group's Computer and Reference Usage Profile</td>
<td>166</td>
</tr>
<tr>
<td>Table No.</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>----------</td>
<td>-------------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>8.1</td>
<td>Components of the HUFIT Toolset</td>
<td>186</td>
</tr>
<tr>
<td>8.2</td>
<td>Vision Module Function List</td>
<td>189</td>
</tr>
<tr>
<td>8.3</td>
<td>Total Usage Errors and Help Enquiries for Evaluated Prototypes</td>
<td>196</td>
</tr>
<tr>
<td>8.4</td>
<td>Expert Evaluation Recommendations (Molich and Nielsen heuristics)</td>
<td>201</td>
</tr>
<tr>
<td>8.5</td>
<td>Total Usage Errors and Help Enquiries for Prototype C Evaluation</td>
<td>202</td>
</tr>
<tr>
<td>10.1</td>
<td>Time Allocation Categorisation Criteria</td>
<td>263</td>
</tr>
<tr>
<td>10.2</td>
<td>Project Time Allocation Summary</td>
<td>269</td>
</tr>
</tbody>
</table>
Glossary

In order to discuss the issues raised by the study it has been necessary to refer to several specialised terms throughout the document. This section aims to clarify the usage of the terms which appear in the document. Definitions of each term are presented below.

<table>
<thead>
<tr>
<th>Term</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activity</td>
<td>A specific group of actions undertaken to meet an individual goal.</td>
</tr>
<tr>
<td>Cycle</td>
<td>A recurring series of phases or processes. The 'modular system' lifecycle model consists of two main cycles. The prototype phase of this model follows three process cycles.</td>
</tr>
<tr>
<td>Development Lifecycle</td>
<td>A series of phases guiding all stages of a product's life from conception through design, development and delivery, including its maintenance and support.</td>
</tr>
<tr>
<td>Development Tool</td>
<td>A software application used to construct a product which can then be delivered to the user.</td>
</tr>
<tr>
<td>Environment</td>
<td>The characteristic features of the workplace within which the development project is taking place e.g. available facilities, time pressures, organisation culture, etc.</td>
</tr>
<tr>
<td>In-house Development</td>
<td>Completion of all stages of the product lifecycle from specification to maintenance on the site of the user group by an individual integrated within the group.</td>
</tr>
<tr>
<td>Information System</td>
<td>A computer based tool designed to provide enhanced access to data in the form of text, numbers or graphics.</td>
</tr>
<tr>
<td>Phase</td>
<td>A set of activities, grouped together in a process model, applied to meet a common aim. Used to distinguish between the various stages of the lifecycle e.g. Confirmation of product suitability through an Acceptance phase.</td>
</tr>
<tr>
<td>Minimal Specification</td>
<td>The description of a required product not exceeding the confidence of the user group.</td>
</tr>
<tr>
<td>Process</td>
<td>A sequence of activities applied to support the goals of a lifecycle phase e.g. formal techniques applied to gather user requirements followed by design iteration within the lifecycle's specification phase.</td>
</tr>
</tbody>
</table>

Table A Definition of Terms
<table>
<thead>
<tr>
<th>Term</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product</td>
<td>A delivered application (or group of related applications in the case of the DAVE system)</td>
</tr>
<tr>
<td>Project</td>
<td>A period of effort relating to the development or maintenance of a product. Used to refer to the development and maintenance of both the DAVE system and its individual components.</td>
</tr>
<tr>
<td>Prototyping Tool</td>
<td>A software application used to mock-up a product for design assessment.</td>
</tr>
<tr>
<td>Scenario</td>
<td>The specific characteristics relating to the development of a particular product. For the DAVE system these included a small accessible user group, development resources limited to an individual and the absence of a firm product concept.</td>
</tr>
<tr>
<td>Small Scale</td>
<td>Used in the context of in-house software development to limit the number of users to a level at which the complete user group can be involved directly in all stages of development.</td>
</tr>
<tr>
<td>Technique</td>
<td>A specific procedure followed to meet the aims of a process e.g. Questionnaire completion to evenly elicit information from the user group.</td>
</tr>
<tr>
<td>Upgrade</td>
<td>Redesign of a delivered product to meet changing requirements or expectations.</td>
</tr>
</tbody>
</table>

Table B  Definition of Terms (cont.)
1 Introduction

1.1 Overview

The thesis presents a lifecycle methodology derived from the longitudinal study of a small scale in-house software development project. The project was initiated by a small group of experts in the field of vehicle ergonomics. The development project's goal was the effective design, development and delivery of a product which was acceptable to the defined user group. This thesis reports the results of a study conducted in parallel with the development project. The aim of the study was to identify an appropriate lifecycle methodology to address the project's goals and to provide guidance for future development within this emerging niche.

A series of software applications were specified, evolved, delivered and maintained in support of the requirements of the user group. These products were grouped as modules of the Database Application in Vehicle Ergonomics (DAVE). Delivered modules include an anthropometric data retrieval tool, a standards database, a vehicle vision arc collection tool and a range of tools and utilities based on the evolution of delivered modules.

The project started with the definition of a guiding lifecycle methodology. Its design incorporated available recommendations relating to the known characteristics of the development scenario. Project features such as close user contact and the technique of high level software development were employed through the approach of design evolution to counter the problems presented by user uncertainty and restricted project resources.

The development project provides a series of development and maintenance case studies based on an increasingly refined lifecycle model. Objective and subjective results gathered for each programme provided the basis for lifecycle iteration. Formal evaluation of each module's usability, usage after delivery, and development time were recorded as direct measures of success for each module and its associated lifecycle configuration. These results are supplemented by qualitative information concerning the process and its environment. The study draws these findings together to raise and discuss the issues relating to small scale in-house development, leading to the recommendation of a lifecycle methodology as a guide for future development.
1.2 Study Background

This study was undertaken as a part time doctoral research project in conjunction with a contract between the Army Personnel Research Establishment (APRE), now the Centre for Human Sciences (CHS), and Loughborough University (LU). A project was agreed to evaluate and support the information technology requirements of the establishment's Vehicle Design and Systems (VDS) group. The involvement of the LU Human Sciences Department was sought due its wide experience in the evaluation of the impact of the computer within the organisational environment. LU provided the VDS group with an individual to work on-site whose primary role was to define and support the group's requirements.

The development and support of computer based systems had become central to the function of many large organisations leading to research into the acquisition process. Continuing advances in computer technology and the consequent spread of the personal computer within small organisations were considered to place increasing importance on the issues of computer based system acquisition within this emerging environment. The characteristics of the small scale VDS project placed special opportunities and constraints for which guidance was scarce. By determining the issues facing this development scenario and tailoring a lifecycle model to match its environment, the study aims to support effective development for a specific but widespread category of potential software customers.

1.3 Thesis Aims

The study's ability to influence the conduct of a parallel development project presented the means of testing a range of arguments relating to in-house software development for a small user group. The thesis aimed to explore the central arguments set out below.

1. Evolutionary prototyping is a feasible approach for small scale in-house software development.
2. Design evolution from a minimal specification based on direct user involvement is an appropriate approach for the development and delivery of an acceptable product.

3. A new lifecycle methodology can be tailored to the characteristics of small scale in-house software development as a guide to subsequent projects.

Over the course of its longitudinal investigation the study explored many issues whilst focusing on the arguments listed above. Issues such as the selection of lifecycle processes to match the characteristics of the study's scenario, and the suitability of a reduced development cycle of continued evolution for maintenance were addressed. These issues are set out and investigated within a series of case studies leading to conclusions that influenced the approach adopted for subsequent case studies. The study's cumulative findings are finally drawn together in a 'modular system' lifecycle methodology that provides a guide to accommodating the characteristics of small scale in-house software development.

1.4 Study Approach

In Eason's (1988) guiding reference for the integration of information technology (IT) within the organisational setting he noted that the development of the tools and techniques presented to support user centred design had been participative, out of necessity. The philosophy behind the Human Sciences and Advanced Technology (HUSAT) Institute's design of these methods was described as 'action research'. The scenario demanded active involvement in the design, implementation and evaluation of their user centred tools to meet the goal of producing products which the users were both able and willing to use. Other cases of this approach to research include Checkland's design of the Soft System's Methodology and the design of Multiview as a guide to system development within an organisational setting (Avison and Fitzgerald 1996). This approach of actively influencing the design, implementation and evaluation of techniques provides a precedent for studying a process without removing it from the context of its environment. Baskerville and Wood-Harper (1996) highlighted the potential for this interventionist approach to appropriately assume a growing role in mainstream information systems research. The study reported here adopts an 'action research' approach by planning, implementing and evaluating a user centred strategy,
systematically controlled through an iteratively defined lifecycle methodology. The study's longitudinal approach allowed the iteration of the lifecycle design based on a series of development and maintenance projects.

Whilst 'active research' offered the only feasible approach to this study, investigation of the system in context weakens the researcher's control over many variables which are central to the traditional quantitative foundation suggested for 'good science'. Strauss and Corbin (1990) provided a strategy for the study of complex systems with the principles of 'grounded theory'. Recognising the influence of diverse interactions in social systems, they suggested that it can be appropriate for a theory to evolve on the grounds of information emerging from its study as opposed to the traditional approach of quantitatively comparing a system against a pre-defined hypothesis. A grounded approach seeks to develop a substantive theory based on a 'rich picture' of the problem area. The researcher's task becomes one of seeking to gather evidence from a broad range of sources in support of an emerging theory as the study progresses. This approach does not reject quantitative methods but instead looks to supplement this data with a qualitative description of the problem area. This approach to research places new demands on the presentation of the findings by relating the gathered information to concepts within a concerted theory which in turn can be reliably compared against observations of the real system.

Conduct of the study from a position of direct involvement in the process under investigation provided opportunities for insight mixed with the risk of partiality. The characteristics of the scenario raised two further issues when considering the contribution of this study to the understanding of its research area. The first was the potential conflicts between the study's research goals and the practical goals of the development project. The second problem was presented by the small sample size dictated by the project scenario which limited the study's scope for drawing generic conclusions. Whilst these features placed obstacles in the way of the study they did not diminish the research issues of determining the feasibility of small scale in-house development and providing guidance for subsequent projects matching this scenario.

The competing roles of researcher and developer had the potential to risk the goals of both the study and its associated project. The development of the product to meet real world requirements demanded that project's priorities held precedence over research
objectives. For example, the parallel implementation of a broad range of specification
techniques could have contributed to the research goals but were excluded due to the
development project's need to efficiently use available time and resources. Despite this
apparent conflict, many similarities remained between the objectives of the study and
the project. Although conduct of the project was subject to political influences, the aim
of objectivity in the investigation of both the product and its development process
remained a common goal. The risks to the study were countered by the adoption of a
controlled approach which drew on formal objective evaluation where possible, and was
supplemented by diverse qualitative sources. By this approach this study has sought to
present a 'rich picture' in support of its theory recommending a lifecycle methodology
based on user centred evolution for small scale in-house development.

The size of the user group on which this study was based raised the important issue of
generalisation when considering its contribution of this area of research. Strauss and
Corbin (1990) provided an answer to this issue in their argument that it is not
necessary to generalise beyond a study's research criteria. The size of user group
remained as an unavoidable feature of the study's research question. Many of the
features of the study's tailored lifecycle hinge on the features associated with a small
user group such as user involvement and developer integration. The constraints of the
project prevented the extension of the development to other user populations. Instead,
the study provides a detailed characterisation of the scenario against which subsequent
developers can relate the features of their own development projects. In doing so it
provides a tailored approach to support the characteristics of a specific but widespread
scenario. Where possible, the study relates its observations to the findings of other
studies allowing the broader consideration of the implications of its findings.

By taking a longitudinal approach, in which issues were addressed through a series of
case studies, the study was able to compare and contrast alternative approaches to the
development process. Meanwhile, common aspects between case studies could be
considered based on the cumulative results of evaluation and observation. This approach
allowed concepts to be raised and investigated throughout the study's time-frame. The
result is a detailed 'modular system' lifecycle methodology, grounded in multiple case
studies, which evolved to accommodate the project's emerging requirements.
Lifecycle activities were selected to meet the individual characteristics of each development project. These activities are presented in the context of process models for each lifecycle phase. The lifecycle model and its component processes were investigated through the study's influence over the design and application of the lifecycle. The evolving model was implemented and iterated in a series of case studies. These development and maintenance projects are presented chronologically. They detail the changes made to the model based on the evaluation of the previous module's development project. The cumulative findings of the case studies are traced through the evolution of the lifecycle model. The final lifecycle and its component processes provide a methodology for the development and evolution of software to meet a user group's stated, emerging and evolving requirements.

1.5 Thesis Structure

Chapters One to Three present the background to the study including an overview of the thesis, the background literature and an organisational outline.

Chapters Four & Five describe and discuss the processes of project specification. This includes requirements analysis and the process of computer platform and tool selection. These chapters reveal the range of information areas for which support might be provided to the user group. The basis for selecting the approach of in-house development is reviewed and the study's approach to the confirmation of development tools is presented. This activity achieved an in-house software development capability and distilled a process model as a guide for subsequent development.

Chapters Six to Eight present a series of case studies following the specification, development and implementation of the DAVE system's modules. The development of each module was guided by the study's recommended lifecycle model. Each project followed the phases of specification, development, acceptance and delivery. The results of formal evaluation during a module's acceptance phase provided an objective measure of the product's suitability. This was combined with direct observations to draw conclusions relating to the lifecycle and its associated process models. These conclusions were applied to the design of the lifecycle as a guide to the subsequent module's development project.
Chapter Nine reviews the demand for maintenance over the study's time-frame and the lifecycle's approach to managing this requirement. It presents the maintenance cycle of the 'modular system' lifecycle and discusses the revised processes applied to formal product maintenance. The added function of computer support and its place within the development lifecycle is discussed.

Chapter Ten presents the usage record gathered for the project's software products and a log of the development effort associated with each module of the DAVE system. The levels and trends in voluntary usage of the DAVE system modules and supplementary applications are discussed alongside the cost in terms of developer effort as a relative measure of success.

Chapter Eleven draws together the concepts raised and explored over the course of the study. A lifecycle framework and a series of process models are presented as a guide to future software projects exhibiting the characteristics of the reported development environment. The limitations of the study are discussed and recommendations are made for future research.
2 Development Project Background

2.1 Chapter Overview

Chapter two describes the organisational environment within which the Database Application for Vehicle Ergonomics (DAVE) system was developed. The structure and function of the Army Personnel Research Establishment is outlined. The position of the DAVE system's user group within the organisational structure of the Establishment is described and the group's internal structure is reviewed. The work programme of the group is considered under the headings of applied ergonomics and vehicle related research. The use of computers within the organisation is reviewed. The chapter concludes with an overview of the establishment's amalgamation with two closely related organisations to form the Centre for Human Sciences.

2.2 The Army Personnel Research Establishment

The support project was initiated by the Vehicle Design and Systems (VDS) group of the Army Personnel Research Establishment (APRE). APRE consulted the Human Sciences Department of Loughborough University (LU) to explore the possible contribution of computer based information handling to meet the specific information requirements of the group. LU undertook to study the group's requirements and to provide support in the acquisition of a tailored information technology solution. Acquisition took the form of in-house, high level software development using a methodology which was adapted to fit the development and operational environment.

The role of APRE was to advise the army on how to optimally support the function of the soldier. The work ranged from the development of psychological tests for selection to the determination of the limits of human physical tolerance in extreme operational environments. By optimising the operational environment it was aimed to improve the comfort and efficiency of the soldier.

APRE emerged from the unification of several separate army medical research groups formed following the Second World War. The establishment has been based on the Farnborough site since 1972. APRE was a civilian establishment primarily consisting
of experts in fields of physiology, psychology, and ergonomics. These fields of expertise were represented by the three divisions of Operator Performance (OP), Applied Physiology (AP) and Personnel Psychology (PP). Supporting these divisions were the Army Occupational Health Research Unit (AOHRU), Instrumentation and Technical Support (ITS), a pool of military subjects for experiments, and administration services (figure 2.1). At the start of the project APRE employed approximately 130 scientists and support staff.

Figure 2.1 Organisational structure of APRE (1990)

2.3 User Group Structure

The VDS group was one of three groups within the Operator Performance division. The group consisted of experts in the field of military vehicle ergonomics. Over the course of the project the size of the group ranged between seven and ten individuals.

Five of the group members, including the contract member, held post graduate qualifications in the area of Ergonomics. One other individual held a post graduate qualification in a related discipline. The junior members of the group were pursuing further education courses allied to the group’s work. Each member had sufficient experience to undertake tasks within the main areas of the groups work programme.
Figure 2.2 illustrates the hierarchy within the group. It was headed by a Grade 7 scientific officer who retained overall control of the work programme. The other members included one Senior Scientific Officer (SSO), one Higher Scientific Officer (HSO), two regular Scientific Officers (SO), an Industrial Placement Student (IPS) and an Assistant Scientific Officer (ASO). These individuals were set as the specified user population for the proposed information tool at the start of the project.

Increasing seniority within the hierarchy led to greater involvement in the planning and direction of work carried out within the group. Higher grades were often involved in the control and reporting of work carried out by lower grade colleagues. Over the course of the development programme the numbers of individuals at each hierarchical level varied due to promotion and transfer. Despite this fluctuation, the overall size and constitution of the group remained similar. A core of five users remained with the group throughout the period considered by this study.

The age and associated level of experience tended to increase in line with this grading system. The position of the contract member within this structure was related to age and experience. The LU contract post was not formally positioned within this structure. In practice, the age and experience of the contract member at the start of the project led to
a level of responsibility in line with the SO members of the group. Over the course of the project these positions changed with promotion, transfer and the increasing level of experience gained across the group.

The group members were located in a series of adjoining offices. The more junior members shared offices in pairs. Levels of communication were high with regular formal meetings and daily informal discussion during regular work breaks. The organisational culture recognised the value of these informal discussion and did not enforce the statutory time allowed for 'break' periods.

2.4 The Contract Post

The APRE had a long-standing arrangement with LU in which it maintained an academic link with the university by employing contract staff on short term contracts. The primary role of these contract posts was to provide additional manpower to assist in the work of the group. The changeover in the VDS group's contract position prompted a re-evaluation of the work programme associated with the position. At this point the group considered there to be a need to direct the core work of the contract towards the support of the group's information requirements.

The individual selected for the contract post held similar post graduate qualifications to the other members of the VDS group. This allowed the individual to provide immediate support to the group's ongoing work programme. The contract individual was recruited based on the same criteria applied to the selection of the group's existing members. The criteria of technical experience in system design and development was considered to be of less importance than an understanding of the core skills of the group's work programme.

The project's work was reported directly to the head of the group rather than through the hierarchical chain followed by the other group members. Despite the group's formal hierarchy, the culture was flexible. The limited workforce demanded that all group members work together as a team. This was particularly evident during field trials in which the whole group was involved in applying the experimental programme. This flexibility extended to include the contract agreement. The head of the group was free to
divert the work of the contract member away from software development to other areas of the group's work programme, as required.

2.5 User Group Work Programme

At the start of the project the group's work programme was split between applied ergonomics and related research. The work was primarily concerned with the design and use of military land vehicles. This expanded to include other areas of ergonomics with the transition of the APRE to the Centre for Human Sciences (CHS), (section 2.7).

The small size of the group and relatively narrow field of work led to a tendency for most members to be involved in each of the group's ongoing projects. In addition to these common work areas, a small proportion of the work held a high security classification preventing discussion with other group members.

2.5.1 Applied Ergonomics

The group's main work area was the application of ergonomic principles to the design of military vehicles and systems. This included a range of activities including the evaluation of existing and proposed vehicle designs, ergonomic consultancy and the specification of human factors input to standards documents.

Ergonomic evaluation carried out by the group centred on the assessment of military vehicles considered for service with the British Army. The type of vehicle assessed ranged from a Landrover to a Main Battle Tank. Vehicles considered were usually either new designs under consideration for procurement or crewstations of vehicles following the retrofit of new equipment. New designs were generally evaluated in a mock-up of the proposed design, (figure 2.3).
The knock-on effect of fitting new equipment within an already cramped environment often had the consequence of poor positioning of the new equipment or unsuitable repositioning of existing equipment. The extreme environment presented by a military vehicle in the field required careful consideration of the crewstation based on an understanding of the operational scenario. The noise, vibration, crew clothing, prolonged activity, stress, etc. all affected the suitability of a design. Each member of the group had past experience of the vehicle environment and operational scenario.

Consultancy carried out by the group broadly covered the area of applied ergonomics. Requests for consultancy generally came from military "customers" and related to a wide variety of work. The enquiries were often telephone based and answered on the same day, depending on the complexity of the enquiry. Telephone enquiries could be taken by any member of the group. A form was completed for each enquiry recording the customer details and detailing their questions. The form would then be passed to the most appropriate individual via the head of the group. Enquiries beyond the group's expertise were passed to the appropriate expert within the establishment. Enquiries for which direct recommendations were not known could lead on to research work to determine the answer. The group's responses were normally based on a combination of existing
reference materials such as Defence Standard 00-25 (Ministry of Defence 1983) and expert advice from individuals.

The group was involved in defining the military's human factors standards, Def. Stan. 00-25, with particular involvement in the drafting of Part 4, Design of the Workspace. This section provided recommendations for both office and vehicle workstation design with particular emphasis on military environments. The group was responsible for ensuring that designs passed to them for evaluation conformed to good ergonomic practice, with particular reference to the standards set out in the Def. Stan. 00-25 documents.

2.5.2 Vehicle Related Research

As a scientific establishment, APRE undertook research with the aim of supporting the efficiency and well being of the soldier. This research ranged from evaluation of the man-machine interface to the modelling of clothing micro climates.

The research programme of the VDS group focused on the design of future vehicle systems. A common part of the group's work was field based evaluation of concept prototypes. Systems evaluated included indirect vision for target detection and a computer based device for the display of vehicle location. In each case the system was fitted to an armoured vehicle and tested in a field environment with representative users. Data gathering techniques applied by the group included observation, video analysis, and questionnaire presentation.

Figure 2.4 shows the prototype map system fitted to a tank for the "Mapcase" field trial. This work investigated the use of a prototype tactical map display and message handling system in comparison with a paper map and radio system (Joyner 1990). As noted earlier, the group's size and the organisation's approach to resourcing demanded that all members of the VDS group take an active role in the trial.
2.6 Computer Usage

The software development project arose from the VDS group's assertion of their requirement for information technology in support of their work programme. The structure of the organisation and its policy towards computer implementation combined with the individual opinions of the user group to influence the course of this project.

2.6.1 Centralised Computing

At the start of the project, computing within the establishment was dominated by the organisation's VAX minicomputer. The VAX was maintained centrally and made available to those requesting a connection via monochrome text terminals. Many people within the organisation resisted the influence of the trend towards computing in relation to their specific job. VAX terminals were often shared rather than given desk space. The
system's main uses were text editing and internal messaging. Other functions such as data processing, database searching and telnet access to external networks tended to be used by a small subset of the organisation.

An increasingly important role of the VAX was to provide a specific group of physiologists with access to the MAPS knowledge based system (Wadsworth 1990). The MAPS (Modular Applied Physiology System) was a collection of functional modules written in FORTRAN which served a range of physiological topics. It provided data retrieval and several predictive models via a multi-level, text based interface.

The MAPS project demonstrates the organisation's acceptance of tailored software development as a suitable approach to supporting a research group's work programme. The apparent success of the MAPS was influential in identifying the VDS project and in backing the approach of tailored software development.

Whilst the initiation of this project was based on a perception of the benefits of specific computer support, these views did not extend to the identification of possible products of a computer based support project. The group's motivation for the project overcame the uncertainties arising from their limited of experience in this area. This shows a parallel with the motivations observed by Isaac and Pingry (1991) in their report of the haphazard spread of personal computers within organisations. An important distinction lies in the VDS group's recognition of the need to investigate how computer based solutions might support their requirements before embarking on a development programme.

2.6.2 Personal Computers

Whilst the establishment maintained control over the central computing facility, in the form of the VAX, the separate sections were free to acquire other computer resources to meet their specific needs. Whilst a couple of groups continued to rely on the VAX as the prime resource for computing, others acquired the new desktop computers such as the BBC computer, and later IBM compatible computers, for both laboratory and office applications. The absence of an organisational computing policy was important in removing restrictions on the project in terms of hardware and software selection. For example, insistence on the use of the VAX network may have led to the premature discard
of the VDS project's software products along with the MAPS when the VAX was decommissioned in 1995.

VDS group progressed from the use of a calculator with a printout for data analysis to early spreadsheet packages on DOS based personal computers. The fundamental benefits derived from flexible data input and analysis ensured their continued use and contributed to the apparently positive view held by the group of the value of computer based tools. The group had acquired a UNIX workstation to run a three-dimensional human model for workspace evaluation. However, the complexity of its interface and the limitations of the model contributed to diminish the real advantage of the tool in comparison with existing methods for workspace evaluation. As a consequence the tool was not widely used.

Examples of graphical user interfaces (GUIs) were starting to appear within APRE with the purchase of Macintosh computers by a few individuals. Whilst GUIs now provide the normal means of computer interaction, text based, command line interaction remained the dominant form of computer interface within the organisation for much of this project's time-scale. The need to learn and remember a vocabulary of commands placed a confidence barrier before many potential computer users within the establishment. Notably the use of computers within the group was limited to its two youngest members.

2.7 The Centre for Human Sciences

Significant changes in organisational structure and management occurred towards the end of the project. On the 1st of April 1994, APRE and the Institute of Aviation Medicine (IAM) were amalgamated and incorporated into the Defence Research Agency (DRA) to form the Centre of Human Sciences (CHS). Restructuring within the CHS resulted in the formation of three departments Physiology, Psychology, and Aeromedicine/Neuroscience, (figure 2.3). Prior to this transition, APRE consisted of 131 employees. The new organisation employed over 300 people at the start. CHS was located in the existing APRE and IAM buildings.

The members of the VDS group remained as a unit following organisational restructuring. The group was expanded and re-titled the Biomechanics and Ergonomics group. The group grew in numbers to around 20 at its peak. This increase represented a
wider range in the fields of scientific expertise associated to the group. Despite the nominal increase in group size the development project continued to centre on the core users of the original group.

The CHS management approach was altered from the original hierarchical structure towards a flatter arrangement. The work of up to 20 individuals was arranged and coordinated by a resource manager. The resource manager controlled the allocation of an individual’s chargeable hours to a range of projects. These projects were in turn controlled by project managers who negotiated the staffing of these projects with the appropriate resource managers. This involved a move away from the small group approach to project management where the head of the group was responsible for all aspects of project and man management. The goal of this new approach was improved efficiency in the allocation and charging of the time of individuals.

In addition to a promised improvement in efficiency, this change in staff management produced several side effects. Individuals became available to join working teams within any area of the establishment. This provided an opportunity to both increase the spread of skills applied to a particular project and to broaden the experience of individuals within the organisation. Associated with this was a broadening of the information requirements of individuals within the organisation.

Although the original members of the VDS group remained intact within the larger Biomechanics and Ergonomics group, their work showed a greater variation between individuals following the transition to the CHS. This reduced the opportunity for informal discussion within the group about specific work areas in which previously all members of the group would have been involved.
Figure 2.5 Organisational structure of the Centre for Human Sciences 1995

2.8 The Small Scale In-House Development Environment

The previous sections presented the broad characteristics of the organisation, the VDS group and their working environment. To consider the feasibility and method of supporting the group's information requirement, it is necessary to relate these characteristics to the strategy of evolutionary development. These characteristics and their implications are set out in the following sections.

2.8.1 Small User Group

- It is possible to achieve direct involvement of all users.

The central feature of the study's scenario was the small size of the user population. As noted earlier, the specified user group did not exceed ten individuals. This holds important implications for the implementation of an evolutionary approach to software development. At the level of specification, it becomes feasible to take specific account of the views and capabilities of each individual. Added to this, induction of the developer within a small user group allows the development of a close working relationship built
on a mutual understanding of the requirements of the group's work programme and the scope of the development project.

2.8.2 Limited Resources

- Tight constraints are placed on the scope of products due to equipment budgets and developer time limitations.
- Support and maintenance demands must compete with development activities.

Under normal circumstances, the cost of development should relate to the value of its product. In the context of small scale development, with resources shrinking in relation to a proposed product's influence, it becomes increasingly important to make optimal use of tools and techniques for achieving the development project's aims. Whilst a large development team might seek to produce a large ambitious application through the division of effort, the individual developer must ensure that the development project's aims do not outstrip the resources available. Delivery of an elaborate application after a long delay risks changes to the stakeholder's original requirements, reducing its applicability and associated use.

The attempt to meet a user group's development and support requirements, based on the effort of an individual, places particular demands on the approach to managing the developer's activities. A framework for addressing a project's requirements is required to avoid a rushed and haphazard approach to meeting and maintaining support for the requirements of the defined population.

2.8.3 Developer Presence on Site

- The developer is able to instruct, encourage and learn from the users.
- There is a need to respond to untimetabled demand for support.

A consequence of in-house development, in which the developer is integrated within the user population, is their accessibility. As noted earlier, this close working environment presents lines of communication, unavailable to the remote developer, on which a mutual understanding can be built. An additional implication rests on the ability of the
user group to directly approach the developer for both maintenance and guidance whenever their requirements arise. In terms of a project's management, these tasks place an added demand on the allocated resources. However, the scope for both encouraging user support for a product and ensuring its satisfactory operation provides an opportunity to counter some of the risks of user rejection such as lack of involvement, and inadequate support (Hirschheim and Newman 1988).

2.8.4 Culture of Peer Group Discussion

- Informal discussion provides a source for specification and feedback.
- The developer is part of peer group during discussions.

The size, structure and management approach of the user group have an influence on its organisational culture. The close working relationship of a small group of experts provides an environment where peer group discussion can help to apply the cumulative experience of the group to a range of current problem areas under informal conditions. A culture of informal discussion provides a rich source of information and feedback to support many of the activities of software development and support. It can provide information ranging from individual preference, as the background to product specification, to detailed information concerning the design and use of current applications. The integration of the developer within this peer group adds a dimension to the discussion by bringing an understanding of the technical components of the development project.

2.8.5 Group Autonomy

- The users are the primary stakeholders for a tailored product.
- The user group are free to use or neglect the system.
- The responsibility and overall control of all the group's activities, including development, rest with the group leader.

The structure of a user group within a larger organisation and the style applied to its management has an influence on the conduct of tailored software development. If overall control of a development project is held at a higher organisational level, a wide group of
stakeholders must be considered. In contrast, a semi-autonomous group of users is able to take more complete control over a product's specification, development and implementation, raising the users' stake in the product. This form of control brings with it a greater influence on a product's eventual design and by association its acceptability. This influence on a product's success does not stop with its delivery at the end of the development process. In the absence of a mandate from the organisation to use the product, the group retain the freedom to choose whether to apply it to their work programme. The product's success, as judged by its level of implementation, rests on the value judgements of the user group as revealed by levels of voluntary usage.

In the case of an individual manager taking overall responsibility for the group, the associated control inevitably influences all aspects of development and support. This was highlighted by McCosh (1984) who reported the association between a manager's qualifications and software success.

2.8.6 Range of Experience in Computer Usage

- Functional and interface design must accommodate a range of computer experience.
- Specification must account for absence of an initial model on which the users might build their product concept.
- Demands for system refinement arise from the users growing experience.

The need to support a group of users who possess a spread of computing experience is a common attribute for many development projects. Whilst we can anticipate a continuing rise in computer literacy amongst the population as a whole, the spread of experience gained and therefore their competence and expectations will continue to require that software designers balance apparently conflicting system requirements. Whilst a simplistic design might support the novice user, its reduced functionality or interface approach might be frustrating for the experienced user. The evolution of a software design drawing on the direct involvement from this range of users offers an opportunity to explore solutions to meeting these demands within an integrated product.

As specification relies on the development of a common concept of the required product, this range of experience serves to further separate the ability of the user group to arrive at an agreed description. Evolutionary development based on prototype iteration
offers a medium for providing the common experience necessary to align the users views in product specification. It is necessary to recognise that the growing experience of the user group over the course of a development project, linked to the close working relationship of the in-house environment will place demands for refinement which must be addressed as they arise.

2.8.7 User Accessibility

- Iterative development is made possible through free access to the users.

An evolutionary approach to design and development matches this small scale in-house environment closely by offering the opportunity to build on informal lines of communication as a guide to the design evolution. The product of this process can then be evaluated drawing on the participation of the complete user population to provide a clear assessment of acceptability. The close interaction fostered by this small scale in-house environment has the potential to keep track of delivered products and emerging requirements to drive the maintenance and continued development activities.
2.9 Conclusions

The structure and function of the organisation defined the environment for both the VDS group's work programme and the project's development programme. The key features considered to be important in relation to the software development project are presented below.

1. The software customer was a small, semi-autonomous user group, able to exercise full control over the aims and approach of the development project.
2. The absence of a rigid organisational policy on computer system acquisition removed potential restrictions from the project's selection between hardware and software alternatives.
3. The group's work programme and associated information requirements were specialised.
4. The impact of the organisation's merger on the development project within the study's time-frame was minor due to the user group remaining intact.
5. The contracted software developer had an equivalent training background to the user group allowing full integration as a group member.
6. Previous computer system acquisition projects within the organisation set a precedent their support for an in-house software development approach.
3 Small Scale Software Development

3.1 Chapter Overview

Chapter three reviews the literature relating to the use of computers by small organisations and the approaches to acquiring information technology to meet the requirements of this emerging user population. The chapter starts with an overview of the growth of the computer industry and the spreading influence of the computer into small organisations. Reports of a high failure rate for computer systems leads to the consideration of factors contributing to their success or failure. Potential sources of computer solutions for the small organisation are discussed. In-house development is considered as an alternative approach in the absence of appropriate commercial products.

The chapter continues by considering recognised approaches to software development. These range from detailed structured methods to the adaptable guidance of lifecycle methodologies. The final section discusses the common phases of software development in more detail. The complementary principles of iterative development and direct user involvement are highlighted as appropriate approaches for in-house development. High level development tools are examined as the basis for a prototyping approach to user centred design. The issue of product evaluation is presented in relation to a user centred prototyping approach to development.

3.2 Background

3.2.1 The Diffusion of Computer Technology

The personal computer market has increased from around 200,000 in 1977 to more than 50,000,000 in 1987 (Nickerson 1992). The numbers are increasing and the rise looks set to continue. Nickerson went on to observe that, during this ten year period, the cost of an entry level computer has decreased by 60-fold. The reduction in the cost of computing has led to a rapid spread in the number and diversity of users (Eason and Harker 1988). Further reductions in processor cost and size will allow computer components to be invisibly incorporated within the physical environment (Weisner 1993) spreading their impact further.
The spread of desktop computers throughout the population has led to new niches for software products. This small scale software development moves the emphasis of the industry away from large computer companies to smaller software organisations (Grudin 1992). The emerging, small scale software industry is rapidly expanding to meet the diverse requirements of the emerging user population.

3.2.2 The Software Industry

Growth in the software industry of the United States and Europe has been in the region of 28% annually (Martin 1988). Market conditions and technological developments have resulted in the appearance of large numbers of small companies to meet the demand for customised products required for specialist markets. Conversely, the rate of progress in the development of new computer hardware technologies dwarfs the progress in the development of software production techniques. It is the development of software which is the limiting factor for the effective application of computer systems in the future (Peled 1987).

3.2.3 Large Scale Software Development

Advances in hardware now make it possible to develop large distributed systems. However, Gibbs (1994) considered large scale software development to be in crisis, with some systems collapsing under the weight of their own complexity. The permutations surrounding the operation of large scale software systems prevent the confident determination of fundamental coding problems. Moves to formalise the development of these systems are seen as the only solution as exhaustive testing of these large software products is not currently possible. Austin and Parkin (1993) found that, despite the heralding of formal methods as the way forward for the production of good quality software, the methods are as yet not widely accepted in industry. They found that use of these methods was mainly due to contract requirement. Problems encountered included a lack of suitable tools and a perceived barrier in the application of the mathematical component of methods. The search for a unified approach which overcomes these problems continues. No single method has been found to offer the full solution. For the present these methods are primarily
applied to large development projects, particularly those involving a security or safety critical component.

3.2.4 Computer Use for Small Organisations

Many organisations have seen a need to embrace technology in order to remain competitive. Information systems can have a significant influence on the success of an organisation (Bergeron and Raymond 1992). The possible contribution to both large and small organisations has been recognised for many years (e.g. West 1975). Computer solutions were initially the preserve of large organisations, able to maintain a central computing department. The reduction in hardware costs and the appearance of diverse, off-the-shelf applications have brought computing solutions within the reach of the small organisation. However, a gap remains when off-the-shelf software fails to meet the small organisation's specific requirements. Filling this gap demands a new category of software, tailored to the needs of the user within the financial means of the small organisation.

The literature relating to organisational software design concentrates on the design and application of systems to meet the requirements of large populations. The impact of these large systems make them highly suitable for investigation. The spread of computers to small organisations increases the importance of this less reported area. The few studies investigating the issues surrounding IT implementation in small organisations, show considerable variation in their use of the term 'small'. Studies referring to small organisations ranged from 3 specified users (Telem 1989) to organisations of under 300 for which the specific user population is unspecified (Delone 1988). Although, in each case they share the attributes of a manageable user set and the cultural characteristics of a small organisation.

Torkzadeh and Rao (1988) considered the implications of the increasing application of computer technology to small businesses, concluding that the reduced cost of the technology presents a lower risk to small organisations embarking on IT implementation. However in a study of ten small companies, Wroe (1986) found that half of the information systems developed for the companies failed. Whilst the effect in these situations may not be dramatic, the cumulative cost to industry must be considerable.
3.2.5 Learning from Large Organisations

Information systems have been used successfully in large organisations to gain a competitive edge but they require adaptation before they can be applied to small organisations (Bergeron and Raymond 1992). Whilst many studies have considered the design and implementation of computer systems for large organisations (e.g. Curtis et al. 1988, Harker 1988, Myers 1990, Trauth and Cole 1992), few have concentrated on small organisations. Whilst large scale development provides a starting point for understanding the process and its implications, it cannot be considered to provide a reliable model for small organisation computer system implementation. It is necessary to consider the specific characteristics of small businesses, in order to understand how best to apply IT to meet their requirements. Small organisations face important challenges in the implementation of information technology.

Small organisations differ in many respects from large organisations, in addition to the obvious disparity in staff numbers. Telem (1989) recognised the unique characteristics of small organisations when addressing the development of management information systems. Small organisations tend to have a simple structure associated with a relative flatness of hierarchy. They often lack data processing expertise, relying instead on external consultancy (Gable 1991).

3.3 Success and Failure

3.3.1 The Risk of Failure

The clearest motivation for IT system acquisition is the perception that the computer offers an advantage over the existing system. However, in many cases the cost of implementation has outweighed the benefits. System failures are widespread and take many forms. Examples range from the high profile failures such as the British stock market computer failure (Kane and Whitebloom 1993) and the collapse of the London Ambulance Service’s booking system (Mullin 1993) to failures which go largely unreported.
As computer software development moves further within the reach of smaller organisations the number of applications developed in-house must rise and with this the number of inevitable failures. The loss of investment associated with the failure of computer system implementation can have a major impact on large organisations. As the proportional investment associated with computer system acquisition for small companies is likely to be higher, the effect of similar failures may be more dramatic. It is therefore important that any organisation considering the acquisition of computer based tools should be aware of the pitfalls and have access to guidelines towards success.

3.3.2 Code Failure

Many of the high profile failures have resulted from systems succumbing to the overwhelming complexity of their program code. The inherent complexity of large distributed systems can prevent the satisfactory determination of the safety and reliability of the system (Jenkins 1993). Corbato (1991) considered failures in complex, ambitious systems to be inevitable. The number of paths through a large program makes it impossible to test all of the permutations. This problem is reduced for the less complex products of in-house software development. The limited programming resources available for small scale software leads to a relatively simple product thus allowing code errors to be detected and corrected before delivery.

3.3.3 User Rejection

A system will fail if it is not accepted by its target population. The broad organisational changes which may accompany system implementation are frequently met with resistance producing a barrier to the effective use of the system. User resistance to computer based information systems and their subsequent failure has been widely documented (Hirschheim and Newman 1988).

Cooper (1994) observed two types of resistance to organisational change. Firstly, organisational inertia, which may cause the failure of system implementation by undermining the design and development of the product. This is often associated with
the imposition of changes without the involvement of the group undergoing the change. At a different level the effectiveness of the product may be diluted during the development process in an attempt to reduce the impact of the system on the organisational culture. The result of this dilution of system goals may be a system which does not meet its full potential leading to complete failure.

Nickerson (1981) investigated the complaints of people who abstained from use or were dissatisfied with interactive computer systems. Failure to match the user's requirements, in terms of system functionality was a common complaint. Nickerson found that even systems built to user defined specifications can fail to meet the users' actual requirements following implementation, as a consequence of shortfalls in the specification process, or a change in organisational requirements. Sources of dissatisfaction included:

- The inaccessibility of the system
- A slow system response time
- Software crashes
- Delays in initial access of the application
- Inadequate understanding of the system due to inadequate documentation

Other individual factors affecting system usage included resistance to change, individuals feeling either inferiority or superiority to the technology or a fear that the system could change the individual's job for the worse.

3.3.4 System Atrophy

Whilst considering the success of decision support systems, McCosh (1984) recognised that systems may atrophy following initial acceptance. In McCauley and Ala's (1992) review of healthcare expert systems, they found that despite the prototyping of hundreds of applied 'expert systems' there has been little interest among the target users. The initial enthusiasm following the early success of systems such as MYCIN and ONCOCIN had led to a wave of development effort. However, the products of this period were largely met with disinterest by the user population. O'Neill and Morris (1989) found that despite a buoyant 'expert systems' industry the take-up of these systems by users may have been affected by an
apparent ignorance on the part of the developer of the user's needs. Two of the
companies interviewed admitted that installed systems had remained unused. O'Neill
and Morris considered this to be an indication of many other systems which have met
a similar fate.

3.3.5 Partial Failure

The clearest indication of failure for interactive computer applications is disuse.
Failure within systems may be complete, as seen in the total rejection of systems by
the user population. At another level, functional components of an otherwise
successful system may be seen to fail by the same criteria of disuse. Eason (1984)
observed that certain features of a banking system remained unused due to
individuals making a series of implicit cost/benefit assessments when they
undertook a task. Rather than trying to optimise the benefits of using the system
they concentrated on reducing the 'costs' by concentrating on a few, often used
functions. The end product may be successful in terms of continued usage whilst
failing to meet its full potential without user training or redesign.

3.3.6 Determining Success

An alternative perspective on system development is taken by Delone and McLean
(1992). In an extensive survey of the literature on information system success
they found that there was no single measure of success but rather many interrelated
measures. They categorised these measures under the six headings of:

1. System Quality - accuracy, flexibility, reliability, etc.
2. Information Quality - relevance, clarity, timeliness, etc.
3. Information Use - enquiry number, enquiry duration, etc.
4. User Satisfaction - overall, specific, etc.
5. Individual Impact - learning, recall, effectiveness, etc.
6. Organisational Impact - productivity, operating costs, etc.

They concluded that information system success was a multidimensional construct
and must therefore be measured as such. As yet no overall model of system success
has been developed to tie together these measures. However, the attributes contained within the list provide a guide to issues which should be considered by the developer.

3.4 Sources of Software

Organisations acquiring software are presented with three main sources.

1. Off-the-shelf software
2. Consultant mediated acquisition
3. In-House bespoke development

The choice of the most appropriate source is linked to the specific requirements and resources of the individual or organisation. If an organisation has a significant investment in existing computer hardware this may define the range of available products. The least expensive source of software application is generally off-the-shelf avoiding the cost of the time and effort associated with software development. A wide variety of mainstream software applications exist to meet the requirements of the main areas of computer supported business and leisure activities. Commercial software development ranges from dedicated software houses to lone programmers distributing software over computer networks for a nominal fee.

3.4.1 Off-the-Shelf Software

The starting point for most small organisations when acquiring computer technology is off-the-shelf software. Nazem (1990) found pre-packaged commercial software to be the major source of computer applications for small businesses. A high level of satisfaction was observed amongst these organisations for the software used. The anticipated lack of technical support provided with off-the-shelf applications was not found to be a source of concern among the users. Conversely, Nazem concluded that the users of this unsupported software may become more proficient in the use of computers.

Commercial software packages often allow a degree of user modification. This may take the form of simple cosmetic effects such as displayed font size or screen colour,
Jørgensen and Sauer (1990). Other packages such as spreadsheets and databases may incorporate macro languages which permit the advanced user to automate a series of functions within the application. Whist this may be considered as a subset of the off-the-shelf source of software, macro programs may take the form of applications in their own right.

Macro generators can translate a sequence of user actions into a repeatable script, such as that for the Microsoft Excel spreadsheet package (Microsoft 1994). This produces an editable text file conforming to the syntax of the macro script language which the user may tailor further to meet their requirements. Whilst the macro languages are aimed at the end user, there is an inevitable overhead in gaining proficiency in the generation of macro scripts. The time, confidence or ability to undertake macro programming may not be present for many individuals to whom this form of application development would benefit. An alternative approach is to automate standard spreadsheet procedures within an organisation by distributing a template spreadsheet to managers without the skills required to generate their own macros. Raymond and Bergeron (1992) found that the success, in terms of usage, of decision support systems in small enterprises was significantly higher when the application was developed by the user, with a spreadsheet package.

3.4.2 Consultant Mediated Acquisition

In the case of organisations which identify a requirement for IT, but lack the internal expertise required to adequately direct the acquisition of a computer system, a solution may be the engagement of an external consultant to mediate in the acquisition process. Depending on the customer requirements this may involve solutions ranging from guidance in choosing and implementing off-the-shelf packages to bespoke software development. Gable (1991) reported that consultant engagement was often necessary for small organisations when acquiring computer systems as their small size restricts the breadth of experience contained within the organisation. However, he found that these organisations tended to overestimate the effect of the consultant on the project, abdicating the responsibility for directing the acquisition process. Although an external consultant may bring specific technical expertise to computer system acquisition, it is the members of the user organisation who have an intimate understanding of the organisational environment in which the
system is to be applied. It is therefore important that the organisation retain active involvement in the direction of the project.

In the case of external consultant development, the technical understanding in terms of system design between the software developer and the normal user population reduces the effective accountability of the designer. The values of the designer can strongly influence the design decisions made during development which in turn determine the effectiveness of the software to meet the requirements of the sponsoring organisation (Kumar and Bjørn-Andersen 1990). They consider it vital for the designer to hold a balanced value orientation for the design and implementation of computer based information systems to be successful. The common emphasis of economic values over socio-technical values can result in the design of inadequate systems.

Soh et. al. (1992) studied the impact of consultants on computerisation success in small businesses. They found that the organisations who engaged a consultant showed higher subsequent usage of the implemented systems. Balancing this, they found that the consultant mediated computerisation projects were less likely to be completed on time and within budget.

The resources required to undertake in-house software development have until recently prevented small organisations from undertaking this solution. Some larger organisations who have maintained a level of in-house IT development and support in the past have turned instead to external sources, termed 'outsourcing'. In a survey of software tools used for development in the health services, Millington et. al. (1991) found that a majority of respondents had a specific policy of buying-in software rather than developing their own. Pickering (1992) reported the concerns held by corporation management at the cost of maintaining the large numbers of diverse computer systems. By transferring the responsibility of maintaining the IT component of the organisation to a contracted firm in place of an internal IT department, these organisations are seeking to limit the associated costs. Outsourcing provides certain organisational advantages, such as a reduction in staff numbers. On the downside this solution presents problems related to lack of control, a loss of expertise and, most significantly from a management perspective, the possibility of no cost reduction in real terms.
3.4.3 In-House Software Development

In-house software development has been the traditional domain of IT departments positioned within large organisations. In this context there is likely to be a gulf between both the experience and physical location of the IT system developer and the user population. The spread of personal computers coupled with the appearance of high-level software development tools makes in-house software development accessible to a wider population. Rockart and Flannery (1983) reported that 'end-user computing' was common in organisations but that its use and control was poorly understood. They use the term 'end-user computing' to refer to software developed and supported directly by the user group. Eason (1988) suggested that this 'do-it-yourself' approach has been made increasingly realistic and cost effective by the development of technology. Risks highlighted with this approach included the lack of technical expertise, underestimation of the effort associated.

Despite these reports Delone's (1988) survey of small manufacturing firms (<300 employees) showed that only a small proportion carried out in-house development. He didn't recommend that small organisations maintain an in-house development capability as such expertise was available externally. This was supported by Ross (1994) who advised against organisations undertaking in-house development for document and text management unless the necessity for this approach and the competence of the development team can be proven. However, Ross notes that with the increasing complexity of commercial text management applications there has been a growing requirement to tailor packages to in-house requirements.

Montazemi (1988) observed that the limited resources of most small organisations prevented them from meeting the expense of employing external systems analysts to guide IT implementation. If an organisation is unable to employ computer specialists to meet their needs, an option may be to develop a 'hybrid' individual within the organisation. The development of technological skills within existing members of the organisation produces individuals with an understanding of both the organisation environment and computer system acquisition. Attainment of these complementary skills would allow the hybrid to direct computer system acquisition from within the organisation.
Palmer (1990) referred to the importance of hybrid managers within the software development industry, through their balance of programming and management skills. The reduced learning overheads associated with high-level software development (Nielsen et. al. 1991) provides an opportunity for a wide variety of individuals to attain the skill necessary for high-level application development. This presents the chance for small organisations to develop software skills internally. Wroe (1987) agreed that new techniques for software generation made the in-house development of tailored applications possible. Wroe pointed towards the potential for adopting an evolutionary approach to development to match the design to the developing company. Warnings are given however concerning the time cost associated with development and the danger of "re-inventing the wheel".

An advantage of software development from within a user group lies in the scope for user involvement. Montazemi (1988) found that user satisfaction was positively influenced by user participation in the development process, and the number of systems analysts present within the firm. The in-house presence of the developer facilitates the involvement of the end user in the system design process. Developing a 'hybrid' member of the user population for the development task provides them with a head start over an external developer in gaining an understanding of the information requirements of the user group. In addition, the associated organisation integration provides a better position to anticipate and handle organisational politics.

Sharp's (1991) study, surprisingly, rejected the view that an understanding of the area of application is advantageous for a developer. Sharp found that whilst a designer's experience in software design supports quality in design, their general knowledge of the application domain had no significant effect on quality when employing a structured design methodology based on Data Flow Diagrams (DFDs). The limitations of sample size, the simplicity of the test applications and the variation between applications may account for the lack of effect shown by variations in the developer's domain knowledge.

The possible influence of an understanding of both application domain and organisational environment afforded by in-house experience cannot be excluded. Giddings (1984) argued that software cannot be developed independently of its
application domain. He suggests that the specific effect of the domain on system success defies adequate prediction. Giddings supports an iterative approach to software development to provide the flexibility required to meet the domain effect.

If the user population's requirement cannot be met by off-the-shelf software the organisation must weigh the cost of commissioning software against their need for the tool. In-house development offers an alternative to the expense of development by an outside organisation. By following an in-house software development project the study presented here considers the implications of this approach.

Sutcliffe (1996) commented that small scale information systems are being developed by the user groups themselves with the growth in fourth-generation languages, application generators and rapid prototyping tools. He suggests that participatory design (Eason 1988) may contribute to these bespoke driven developments.

3.5 The Software Development Process

Between the determination of a need for a piece of software and fulfilment of that need lies the task of software development. The problems facing the developer range from the choice and application of development tools to the specification and design of a usable product. Guidance for the developer can be found in many forms, from the basic list of recommendations to detailed methods which control every aspect of development.

Meeting the user's stated or implied needs requires the consideration of numerous factors ranging from hardware specification to organisational culture. There are a wide range of approaches to design and development. No single approach has been isolated as the correct method for ensuring that the features and characteristics of a software system satisfy all of a system's needs, stated or otherwise (Robson et. al. 1991). The diversity of ways in which computer systems may be applied and range of requirements between the various user populations prevents the unification of available approaches into a single, agreed, 'best practice' for computer based system development. Instead, the task facing the developer is finding an approach which matches the particular characteristics of the development project.
3.5.1 Development Methodologies

The traditional data processing approach in which systems are designed and developed in isolation from the users has declined in recognition of the need to include the users in the development process. One solution has been provided by structured design methods in which an explicit contract is set out between the developer and the users. These methods are prescriptive in their guidance of the developer through all stages of design and development. An example of these methods is the Structured Systems Analysis and Design Methodology (SSADM), which was prescribed as the mandatory standard by the United Kingdom's central government in 1983 (Edwards et. al. 1989). The structured approach provides a traceable sequence of detailed development procedures to allow close management of software development projects. Boehm-Davis's (1992) comparison of structured methodologies concluded that, the guidance provided may lead to reductions in design time and product complexity.

The large scale software development industry has been struggling with the formalisation of the development process to avoid the failure of complex systems. Within these large structured methods, direct consideration of the requirements of the actual users remains a small part of this formal structure. Although users offer a source of information for system designers within structured methodologies such as SSADM they often do not influence key decisions in the design process. Consequently the system may fail to reflect the user and organisational requirements. Attempts have been made to revise structured methodologies to incorporate human factors issues from an early stage in the development to overcome this problem (Damodaran and Beck 1988, Lim et. al. 1990, Sutcliffe and Wang 1991). Whilst showing promise for drawing attention to human factors considerations within large scale development projects, issues remained to be resolved concerning the integration of human factors within these methodologies. Damodaran and Beck note the difficulty of transferring the Human Factors knowledge required for integration to the individuals applying the structured methodology.

Despite the recognition of a need to guide software development amongst software developers there has been a limited spread of structured methods. Raghavan and Chand (1989) attribute this to the association of structured methods amongst
software developers with complexity, incompatibility with normal development methods, and a perceived threat to creativity. Robson et. al. (1991), in a survey of systems analysts, found that these methods were often applied partially or in an unstructured manner. No single method was considered to be better than all of the others for producing successful products. Part of the problem facing the acceptance of structured software development methods may lie in a basic conflict between the need to manage the process and the essentially creative process of software development. Structured development methods seek to provide the framework necessary for the development process to become an engineering process forcing it away from the traditional craft based nature of programming (Bott 1988, Wroblewski 1991).

An alternative approach to defining a methodology as a guide for developers is provided by the concept of participative design (Eason 1988, Macaulay 1996). A methodology developed around this principle was offered by Mumford and Weir (1979). Their Effective Technical and Human Implementation of Computer Systems (ETHICS) method, concentrates on the issue of organisational change. It produces a plan for meeting the future needs of the organisation and its users by analysing a system from both social and technical perspectives. Eason (1988) recognised the goal of integrating social system design with technical design. However, the need for expertise in its application and the differences in the separate design processes was considered to account for its limited implementation.

3.5.2 The Lifecycle

The previous section considered the methods used to direct the developer through the development process. Whichever method is chosen for development, the product passes through a sequence of phases from conception to decommission amounting to a lifecycle model. The software lifecycle model is a primary tool in the management of time and resources, providing a strategy for the software development project (HMSO 1993).

Fox (1982) presented the software lifecycle in its simplest form, breaking it down into the phases of development, use and support, (figure 3.1). Fox noted that the development phase was the most troublesome. Consequently lifecycle models tend to concentrate on the processes which make up this first phase.
The stages through which a development project is passed can be used to categorise its development lifecycle. Traditional structured development projects have tended to adopt a linear approach, moving through the sequential stages of specification, development and implementation. This 'Waterfall' lifecycle provides a clear structure for the management of projects (Royce 1987), (figure 3.2). This engineering approach to project management presents problems for the development of software products due to inflexibility. Recognition of these limitations has led to the pursuit of other lifecycles which allow the management of change within the model.

James (1991) suggested that a linear lifecycle such as the Waterfall model would be suitable for developing systems with well understood capabilities such as payroll systems in which the requirement is to identify the components and interactions of the system. However, a flexible lifecycle such as evolutionary prototyping would suit the development of systems involving less predictable interaction, a key characteristic of humans. In these projects, testing the system with a representative population may be the only way to ensure that the design meets the requirements of the users. This 'user centred' approach to system development recognises the influence of user diversity and the operating environment on performance.
Harker et al. (1992) observed that large development projects usually follow a linear design model as compared to small development projects which make more successful use of iterative methods. In all of these cases, there remains a level of uncertainty within the design which must be accommodated rather than fought against.

One approach to including prototyping within the classic sequential lifecycle is the Spiral model (Boehm 1988), (figure 3.3). By incorporating windows for product testing and modification within a structured lifecycle, flexibility is added to the process without removing the means for setting milestones.

The spiral model formalises the iterative approach, controlling the points at which design changes must be determined and implemented. Whilst it is possible to complete a project with just one iteration of this requirements capture and analysis process, Chatzoglou and Macaulay (1996) found that over 80% of the 107 projects studied completed two or more iterations.
Each lifecycle provides a guide to the processes of software development from conception to implementation. Choosing a lifecycle will shape the programme of a development project. At a functional level the proposed phase of a lifecycle and their sequence will control the ability of the development to reach its goals. The choice of a development approach must consider the characteristics of the project. Howell (1992) noted that there is no mandatory requirement to use a particular development method exactly as described. He stressed the tailorability of development methods and suggests customisation of one to meet the particular needs of your development project. Howell provided an example in which the waterfall method was modified to incorporate rapid prototyping, (figure 3.4).
3.6 In-House Software Development

3.6.1 User Centred Development

Software has moved from its origins as something both developed and applied by computer scientists to support a discerning user population. With this transition, the process of its development has recognised the necessity of user input. Kling (1977) reported the shift in focus from the technical process to the requirements of the user for software development. Involvement of the users and consideration of the organisational setting was seen to offer safeguards against the idiosyncratic influences of a remote developer.

The involvement of the user did not fit comfortably with existing development processes representing a further impediment to delivering a product on time and within budget. Software development continued to be constrained by financial considerations of which the allocation of time was a prime consideration. In the early seventies Weinberg (1972) demonstrated a trade-off between software development time and the consideration of usability features. Programmers, when asked to develop software to complete an identical task, but given the different development criteria of ensuring a minimum development time, minimum storage space, etc., neglected the usability criteria such as output readability.
For software to accommodate the requirements of the individual and organisational requirements of the user population new approaches were required to the development process. Kelley (1984) recommended direct user involvement in an iterative development process both as a way of providing input to the design and as a method for facilitating the process of implementation. This approach was considered to be particularly important for naïve users. In addition to removing a source of design information, failure to include the users can lead to their alienation. Mantei and Teorey (1988) reported that user involvement can help to avoid the outright rejection of a system. However, they warn about the potential for over design of the system in which user benefits are not reflected in the extra development cost. They considered the benefits of user involvement to be real and accountable but suggested that there was a point in terms of user population size at which the costs of user testing and task analysis outweigh the benefits offered by these methods.

Eason (1988) established user centred design as a concerted approach to development with a detailed consideration of the effective role of the user in the development process. A range of tools developed by the Human Sciences and Advanced Technology (HUSAT) Research Centre are reported. These tools were provided as guides to effective socio-technical design allied to the principle of iterative design. These tools were designed as a guide to projects implementing information technology, combating the traditional technically driven approach. They address activities including the selection of a design process, system specification, meeting the needs of the user, implementation and user evaluation within the organisational setting.

3.6.2 Specification

The starting point of each of the lifecycles considered was a form of product specification. However, arriving at a specification for what might be a high level concept is not a straight forward task. Spence and Carey (1991) found users to be unable to adequately set out their requirements in a specification document without prior experience of what the product might do. Harker et. al. (1992) considered the traditional view of system specification, as a task of capturing a static set of requirements, to be inappropriate. One of the common problems identified by developers working within this sequential approach to development was the
instability of requirements. Cherns (1976) highlighted the need for flexibility in the socio-technical design process, presenting the concept of minimum critical specification in design.

A user's perception of a software system may change following demonstration of the product (Hekmatpour and Ince 1987). Following the traditional sequential model, demonstration of the system to the user takes place late in the product lifecycle. Users require some experience of a proposed software system before they are able to specify accurate and stable requirements (Luqi 1989). Whilst the normal starting point of all software development projects is the specification of the product, the indication is that users are not in a position to provide a rigid specification.

Smith (1991) highlighted the risk of the following assumptions within the software development process:

1. All requirements can be analysed and understood before development.
2. Requirements will remain stable throughout the development process.
3. Users fully understand the technical documentation presented to them.

Smith suggest prototyping as an appropriate approach to avoiding these assumptions. He notes the risk of the prototyping approach of not reaching the expected product but suggests that this may be offset by the reduced cost and complexity of this approach.

Boehm et. al. (1984) when comparing prototyping to direct specification, found that prototyping was faster, generated less code and produced a more usable product than pre-specification. Specification however, produced a more coherent design which was robust and included improved functionality.

Harker (1988) considered prototyping and simulation to be important tools in the development of large software systems. The importance of ensuring user involvement in the design process was stressed, as was the necessity for hands-on experience by the users. Simple demonstration of the prototype can allow the consultation of several users at the same time. Hands on experience for all users was considered preferable as a multi-user presentation provides a less realistic
impression of the characteristics of an interactive system. Harker noted that selection of a representative group from the user population and provision of a realistic task scenario was vital when assessing user interaction with the system. Presentation of a prototype coupled with evaluation was presented as a useful source of data ranging from objective measures of speed and accuracy to subjective views and opinions.

Macaulay (1996) presented the case for a requirements engineering approach in which techniques are chosen from a developer's repertoire to meet the characteristics of the scenario. Techniques such as co-operative requirements capture, prototyping and co-operative evaluation, amongst others, are presented as methods of determining and meeting product requirements within the framework of an iterative development project.

3.6.3 Design and Coding

3.6.3.1 Prototyping

A prototype is a "Software system that simulates or animates the structure, functionality, operations or representations of another system" (The Open University 1990). There are various forms of prototyping available. The main types are:

1. Rapid Prototyping - Quick creation, discarded following evaluation.
2. Evolutionary Prototyping - Evolution of the prototype into the product.
3. Incremental Prototyping - Prototyping in sections, based on an overall design.

The boundary between these prototyping forms can be indistinct. An evolutionary prototype may be discarded in place of an alternative design. Equally, components of a rapid prototype may be included in a subsequent product. A common feature is that a prototype should be developed quickly and cheaply.

Gomaa (1983) found rapid prototyping to be an effective method of determining user requirements for a software product. The problems uncovered by the technique justified the expense of the rapid prototyping phase of the project. In an industrial application, Jasany (1990) found the technique of rapid prototyping freed the software developer
from programming problems allowing them to concentrate on creating solutions to application problems. The effect was to enhance the rate of software development.

A prototype can range from a paper representation of a proposed interface screen to a fully functional application. In each case the prototype must provide the user with an impression of what can be provided and allow the user to direct design changes to meet their requirements before delivery. At a low level, the simple presentation of a non-functional interface to the user population and eliciting feedback provided a source of recommendations for improvement (Candy and Edmonds 1988). Nielsen (1990) found that a HyperCard mock-up of a videotex system tended to focus the evaluators on the major usability problems more than the equivalent paper mock-up. It was noted that the types of problem discovered using the two methods differed greatly leading to the recommendation that both methods should be used.

Evolutionary prototyping and incremental prototyping cross the line between being a method of specifying a piece of software and the actual development of a deliverable product. Evolutionary prototyping offers an advantage over incremental prototyping by responding to changes in requirements as they occur in the development project (Hekmatpour and Ince 1987). A danger associated with the development of the evolutionary prototype was that an unsuitable feature may become embedded in the design.

3.6.3.2 High-Level Software Development

There are a wide range of software development tools, catering for a wide range of application requirements and programming experience (The Open University 1990). The advent of high-level languages offer a simplification of the task of programming, debugging and maintenance. The cost of this simplification lies in the limitation of design details and increase in code size and a slower execution rate. Brooks (1982) considered the advantages in productivity and debugging associated with high-level programming to overwhelm the objections of code size and execution rate. Continuing advances in computer hardware serve to diminish the issue of execution rate. These new development tools bring software programming within the reach of small organisations.
Pressman and Ince (1994) pointed out that for small applications it may be possible to move directly from the requirements gathering step to implementation using a non-procedural fourth generation language (4GL). They describe 4GLs as tools which allow the software developer to specify some characteristic of software at a high level. They predict that as the demand for software continues to rise during the mid to late 1990's 4GLs will contribute to an increasing proportion of the software produced.

Because of their relative simplicity many of these high level tools have been categorised as prototyping tools. They offer quick construction of interactive applications ideal as a pre-coding prototype for large software projects. There are a wide range of suitable prototyping tools now on the market for the personal computer. The low cost and ease of use make them accessible to small organisations (May et. al. 1991).

Building on the card based concept of Xerox Parc's 'NoteCards', high-level 'prototyping' tools such as HyperCard and SuperCard have filled a niche in the area of end user computing. These environments have been designed as modular construction sets enabling novice programmers to create applications for personal information needs (Cisler 1988). HyperCard is an object-oriented tool with restricted functionality. It allows the user to produce functional applications early in their learning curve for the application. HyperCard is a tool designed to simplify the construction of functional applications (Johnson and Johnson 1988). HyperCard has become recognised as a prototyping tool due to its support for rapid construction of functional interfaces (Open University 1990).

Simply categorising these environments as prototyping tools fails to recognise the extent to which they are used to construct implemented applications. A measure of this is the extensive archives of HyperCard shareware and freeware applications available on the 'internet' computer network. The literature contains numerous examples of card based applications ranging from a foetal heart rate digitizer (Coleman et. al. 1993) to risk assessment and route guidance for hazardous materials (Beroggi and Wallace 1994). The development of many of these products has been undertaken by amateur programmers targeting a wide variety of special application requirements. These public examples are an indication of the existence of a much larger collection of private applications. Where requirements exist for a particularly fast processing rate, the HyperCard environment has been used to its advantage in creating the front end
interface to applications. The functionality of the application is coded in a lower level language for speed but controlled by a modifiable HyperCard front end. Examples of this include a museum information system (Poulter et. al. 1993), the University of Sheffield Information Resource (USHIR) (Nicolson 1990) and the MEMOIRS personal database system (Young et. al. 1990)

The role of these high level software development tools depends on the size of the project. For a large development programme these tools could be used to prototype an interactive representation of a proposed design for user testing. In this case the tool helps the design stage of a traditional lifecycle. For a small development programme the functional prototype might meet the full requirements of the users. This simplifies the lifecycle by removing the need for the coding phase. In both cases the development programme has the opportunity to guide design through prototyping.

3.6.4 Evaluation

Each of the lifecycle approaches considered requires product evaluation at some point in the cycle. In the classic waterfall approach formal evaluation takes place at the end of the project to confirm that the developer has conformed to the pre-defined specification document. The spiral model demands product evaluation at regular intervals throughout its development. The rapid prototyping model includes prototype evaluation as an inherent part of the technique of prototyping prior to formal coding and delivery. Whilst these approaches choose different points at which to assess the product, evaluation is presented as a necessary part of each model.

Howard and Murray (1987) identified five distinct approaches to evaluation. These are:

1. Expert based - An individual's domain expertise used to evaluate a system.
2. Theory based - Mapping relationships between a user model and a system model.
3. Subject based - Subjects completing set tasks under experimental conditions.
4. User based - Personal evaluation by the user of the system.

Each of these techniques needs a product model for evaluation to take place, with the exception of market based evaluation which uses the in-service product. Expert
evaluation offers a quick technique which is adaptable to any stage of the development project. The design and use of the models required to undertake theory based evaluation require considerable expertise making it less suitable for in-house development. Subject based evaluation is the most widely reported technique and forms the basis of usability laboratory evaluation. User and market based evaluation tend to take place at the end of a development project. The information from this source is largely subjective.

Howard and Murray reported that evaluations tend to influence subsequent development projects as opposed to the system under evaluation. Prototyping within the development process provides a means of using evaluation to improve the product being evaluated. The adoption of a prototyping approach to software development lends itself to the user based and expert based approaches to evaluation. In-house development particularly matches the requirements of user based evaluation by providing the potential for close contact between the developer and the users throughout the development process.

Most of the evaluation approaches considered rely on subjective responses from various sources. Subject based evaluation seeks objective measures of performance. As part of the development procedure a measure of performance provides an assurance that the design meets the appropriate operating requirements of the proposed product. Scudder and Kucic (1991) recommend that multiple levels of performance should be assessed in the determination of information system productivity. A single measure is easier to apply but they warn that the results may be misleading. The acceptable levels of system performance may lie in execution time, number of errors, etc. Interface suitability may relate to issues of response time and conformity but strays into the area of aesthetics and intuitiveness for which acceptance criteria is harder to set.

Nielsen (1994) reported an increase in the corporate approach to usability evaluation of products as reflected by the set-up of specific usability laboratories. These laboratories generally take the form of rooms set-up for the observation and recording of user interaction with a product prototype. There has been an increasing trend for the development of usability laboratories for the evaluation of office based systems. This follows an acceptance of the importance of environmental factors on a user's operation of a system. By recreating a normal operational environment the aim is to ensure that the users interact with the system under evaluation in a normal manner.
Palmiter et. al. (1994) took the approach of presenting the proposed product to the user in their workplace as a means of achieving feedback from the actual user population. Small scale software development neither justifies or requires the construction of a usability laboratory. The usability laboratory attempts to maximise ecological validity through simulation of the product’s normal operational environment. This may be bypassed in small organisation by evaluating the product within its actual operational environment. The advantage of evaluating within the real environment is balanced against the loss of experimental control associated with evaluation within the real world.

Another approach to gathering usage information is background logging. Teubner and Vaske (1988) considered that software monitoring provides the most suitable approach for the collection of behavioural data. It is unobtrusive and can provide accurate data for large samples over a long period of time. This form of integral action monitoring can be built into functional prototypes for evaluation or implemented systems as a means of gauging software acceptance. Nickerson (1992) raised the concern that this form of monitoring may be exploited as an assessment of work levels for management purposes. The consequence of this perception amongst the user population could result in rejection of the system.

3.6.5 Maintenance

System implementation does not stop with delivery of the product. Fox’s (1982) simplified model of the software lifecycle presented the concept of maintenance as a period of continued development running in parallel with the use of the product, (figure 3.1). He noted that this support phase can account for more than 50 percent of the total lifecycle cost for large programs. The continued development tasks listed by Fox include:

1. Adding New functions
2. Updating functions
3. Updating Equipment
4. Correcting Errors
It is notable that the correction of errors was listed as the fourth task in the list. The larger maintenance task was based on a demand to upgrade the product over time indicating changes in requirements. Arthur (1988) reported that in a typical environment less than 10% of maintenance tasks relate to fixing defects. The majority of the maintenance task took the form of software evolution. This requirement may change due to the realisation of a better way of completing a task, or the possibility of using equipment unavailable at the time of original development.
3.7 A New Methodology for Small Scale In-House Development

This chapter has reviewed the spreading influence of the computer. Allied to this spread, emerging tools and techniques have brought the option of pursuing a technological solution within the reach of small organisations. An overview of the risks associated with both large and small scale development activities was presented. Despite these pitfalls, in many areas effective computer utilisation has been seen to be a prerequisite for success. The rise of high level development tools offers an increasing scope for software design by individuals with no previous development experience, allowing them to meet specific processing requirements of small user groups in a manner that might otherwise have been insupportable. However, little direct guidance is available to support this emerging group of potential software developers. The need for guidance does not end with the inexperienced developer. Any approach adopted would need to account for the implications of a range of computing experience amongst the user group on their relative contributions to the development process.

The review highlighted important principles for software development such as design iteration and direct user involvement. Small scale in-house development presents a close working environment within which direct interaction between the developer and the users has the potential to contribute to all stages of product development and maintenance. However, the literature relating to the process of software development has concentrated on the support for the development of large scale software systems. These projects are characterised by the involvement of a wide range of stakeholders, often relying on feedback from representatives of the user population rather than the user group themselves. Their scale allow them to support the cost of learning and implementing detailed prescriptive methodologies such as SSADM. This contrasts with the tight resources of small scale development that make the adoption of these methods prohibitive. The alternative lifecycle models, such as the spiral model, represent important development principles including iterative design and assessment whilst continuing to define milestones within a process which was designed to meet the characteristics of large scale development projects. The particular features of small scale in-house development and the approaches suited to this environment demand the design of a new lifecycle methodology tailored to this environment.
4 Specification

4.1 Chapter Overview

Chapter four discusses the selection of the project's initial lifecycle, its approach to the
determination of the stakeholder's requirements and the selection of appropriate
technical tools. The issues addressed include the selection and implementation of
specification techniques to support the project's goal of product delivery within its
limited resources.

A framework based on a sequential lifecycle model was selected as a guide to the project.
The study aimed to amend this model based on the project details revealed by the
specification activity. The specification phase was separated into the two parallel tasks
of requirements analysis and the technical specification. Requirements analysis followed
both formal and informal techniques to investigate the group's requirements and their
use of information. Formal processes including user interview, questionnaire
completion, analysis of the organisation's records and voluntary work logs are
considered as methods of determining information demands. These techniques are
contrasted with the informal techniques of developer induction and free discussion as
appropriate methods for requirements analysis. The parallel task of technical
specification reviewed the project's options in relation to hardware and software
development tool selection. The issues considered during this process are presented and
discussed.

The background information gathered by these parallel analysis activities provided a
basis for product definition. The project's user centred approach was reinforced by
taking a formal group meeting approach to product definition. The meeting constructed a
prioritised development list of flexible product definitions as a top level guide to
subsequent development. Issues including the detail of definition, the contribution of the
preceding activities and the role of the individuals involved in the meeting are
considered.
4.2 Initial Lifecycle Framework

The project was initiated by the VDS group based on their asserted requirement for computer based support for their work programme. The starting point of the project was an investigation of the group, their work programme and the organisational setting to determine how a computer system might be applied to support them.

A review of the VDS group’s work programme, (section 2.4), showed that they were working in a narrow field which depended on both personal expertise and access to specific information. Despite the risks associated with bespoke software development, many of the group’s specific work areas could not be supported by the available commercial software. Although commercial software relating to the group’s work started to appear during the course of the project, this software did not overlap with the group’s tailored software and was acquired separately. The rise in software tools from other sources contributed to the overall increase in the group’s use of computers over the course of the project.

A common feature of the methodologies and lifecycles reviewed, (section 3.5), was the use of specification as a starting point. After specification, they differed in their approach to managing the demands of development and delivery. In order to manage the VDS project, a lifecycle was adopted. In the absence of a model clearly suited to the project’s development scenario, the Waterfall Model, (figure 3.2), was chosen. Although this simple, sequential model represented a traditional approach to software development, it provided a clear structure against which the management of the project could be planned. The intention was to tailor the lifecycle model to the project characteristics, as determined through the specification phase, (figure 4.1).

Whilst providing a structure, the lifecycle model did not guide the choice of techniques to meet the aims of its phases. Rather than attempting to specify a product from the start, an analysis of requirements and technical scope was undertaken as prescribed by a socio-technical approach to development (Cherns 1976). In the context of the VDS development project this was considered to divide into two distinct activities which could be undertaken in parallel. The first was the analysis of the requirements on which the specification of the product could be based. The second task resulted from the absence of an organisation standard for the computer platform. As a consequence, it was
necessary to assess the capability of potential computer tools to meet the requirements determined by analysis. The results of these parallel activities were aimed at providing a product definition for subsequent design and development, (figure 4.2).

Figure 4.1 Development Project Lifecycle Model

Figure 4.2 Specification Phase

The following sections consider the parallel activities of requirements analysis and technical specification in greater detail. They discuss the selection, application and results of the techniques chosen to meet the aims of the specification phase.
4.3 Requirements Analysis

4.3.1 Overview

The requirements analysis process aimed to gain a more detailed understanding of the group, its role and the environment under which it operated as a means of identifying appropriate information areas for support. The techniques applied in support of this aim are outlined in tables 4.1 and 4.2. The chosen techniques are classified as either formal or informal in terms of the relationship between the user and the developer.

<table>
<thead>
<tr>
<th>Informal Techniques</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developer Induction</td>
</tr>
<tr>
<td>User/Developer Discussion</td>
</tr>
</tbody>
</table>

Table 4.1 Informal Techniques for Requirements Definition

The formal methods, (table 4.2), provided a quality audit trail which was important from an organisational perspective. However, the close working relationship between the developer and the users allowed the formal methods to be supplemented by the potentially rich informal sources of information. The informal techniques applied are outlined in table 4.1. Whilst the formal techniques were completed in sequence, informal activities were, as the category suggests, unstructured.
Formal Techniques

<table>
<thead>
<tr>
<th>User Interview</th>
<th>Structured interview, formalising user involvement and assessing user expectations of the proposed system.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Work Programme Analysis</td>
<td>Analysis of the organisation's records of the group's time allocation codes to confirm main areas of group activity.</td>
</tr>
<tr>
<td>Personal Work Log</td>
<td>The detailed recording of each individual's work practice and regular information sources.</td>
</tr>
</tbody>
</table>

Table 4.2 Formal Techniques for Requirements Definition

4.3.2 Developer Induction

As noted in section 2.3, the developer's training background was closely related to the user group's area of expertise. This allowed the developer to work closely with the members of the VDS group on their regular work programme. The developer's direct involvement in the group's work programme was undertaken at the cost of direct effort in relation to the project. Justification for this balance was made on two levels. The first was the perceived contribution of work participation to the developer's understanding of the group's work practice and information requirements. A more straightforward cause was the organisational need to fill a shortfall in the group's resources in order to meet work commitments. This shortfall arose due to the organisation's hierarchical 'group' structure. High priority tasks were met from the group's immediate resources before approaching other parts of the organisation for support. Restructuring during the formation of the Centre for Human Sciences addressed this problem of resourcing.

Throughout the project, the 'developer' alternated between the development effort and regular work programme activities including vehicle evaluation, report writing, telephone based consultancy and research, both in the laboratory and in the field (Beagley 1996a, 1996b). In addition to this broad participation in the work areas of
the group, the developer gained experience in the administrative procedures associated with the function of the group. These ranged from form filling to document searches. Redirection of developer time was at its greatest during the early stages of the development project with the intention of supporting the induction process. By the later stages of the project the developer had established a good level of communication and broad understanding of the group's work. Consequently, the periods of interruption towards the end of the project presented fewer benefits to offset the reduced development effort. However, transfer of developer effort to the regular tasks of the group's work programme was replaced by technical support to the work programme, based specifically on the software development skills acquired over the course of the project. Supplementary applications were developed outside the aims of the original project in support of the group's research programme, (appendix H).

4.3.3 Informal Discussion

Direct involvement of the user population in specification, development or evaluation normally involves formal points in the cycle at which the user is consulted. This is particularly necessary for projects where the product is developed remotely for large user populations. The project's 'in-house' location allowed close contact between the developer and the user population on a daily basis. This permitted unrestricted discussion in addition to any formal contact built into the lifecycle. This was not seen to remove the need for formal user involvement but rather as a supplement to formal methods, bringing closer the possibility of a truly user centred project.

At the start of the project, the VDS group expressed uncertainty as to the form an information support tool should take or which specific area should be supported. This is reflected in their formal interview responses, (section 4.4.3). The group showed a growing understanding of what could reasonably be achieved given the project's resources over this period. This was apparent at the time of the formal meeting, during which each member of the group was able to provide a meaningful contribution to the discussion. This growth in technical knowledge within the user group can be attributed to the high level of individual involvement in the project.
4.3.4 Background Interview

The first formal technique applied in the sequence of requirements analysis activities was the structured interview of all members of the group. This technique was chosen at this early stage as a method of evenly consulting the complete user group.

The interview was carried out at the convenience of the individual members of the group. The interview sheet, (appendix A), was designed to determine the personal time allocation, computer experience, requirements, preferences and expectations of a future system of each individual member of the specified user group. It was structured to allow comparison of responses across the group. Their responses are summarised in table 4.3. Although structured, the interview approach allowed discussion to extend beyond the set questions.

The interview results were used to generate a report of the initial views of all members of the specified user group. Most of the group members were involved in each of the listed work categories. This provided a starting point for the consideration of suitable areas for support. Their expectations of a possible system were broad and non-specific, referring to general provision of vehicle related human factors data. The group’s experience of computers and information systems ranged greatly with the majority having very limited experience. The general lack of prior computer experience was reflected in the features considered to motivate use which centred on ease of use rather than functionality. As expected, all members of the group expressed a preference for paper as a medium for presentation. Acceptance of a future system would need to overcome the group’s preference of paper over screen presentation through the potential added value of a computer based system over a paper source.
<table>
<thead>
<tr>
<th>Question</th>
<th>Response</th>
</tr>
</thead>
</table>
| 1. Areas of Work                              | - Vehicle Assessments  
- Report Writing  
- Administration  
- Research  
- Human Factors Consultancy  
- Task Analysis |
| 2. Computer experience                        | - None  
- APRE's Modular Applied Physiology System  
- Oracle SOL |
| 3. VDS system expectations                    | - Reference for vehicle related ergonomic data  
- Retrieving Human Factors Data  
- Integrated subject areas  
- Comparison of info. from different sources  
- Combination of HF info. and Admin. support  
- Database of internal files |
| 4a. Usage Motivators                          | - Includes data beyond personal knowledge  
- Includes current data  
- Easy to understand |
| 4b. Usage Demotivators                        | - Unclear interface  
- System Complexity  
- Inaccuracy |
| 5. Interface preference                       | - Mixed Graphical / Textual representation  
- Direct manipulation |
| 6. Input Device                               | - Keyboard plus Mouse or Touchscreen  
- Avoid pen input  
- Avoid speech |
| 7. Paper/Computer preference                  | - Paper (preferred by all)  
[portable, clear, stable, good contrast] |
| 8. Other points.                              | - Resource for VDS group use only  
- Would like a portable version  
- Should be maintainable by group members |

Table 4.3 Structured Interview Responses

4.3.5 Time Allocation Survey

Having established the group's background and expectations through user interview, the aim was to characterise the group's activities. The technique applied was a questionnaire to survey the individual group members' perceived time allocation. This technique was considered appropriate due to the specific nature of the enquiry. It was
possible for the developer to set out specific and comprehensive questions prior to the survey as compared with the more open enquiry conducted through user interview.

The survey was applied as an individual exercise to gauge the spread of tasks within the group. Each subject completed a survey sheet in which they estimated the proportion of time they allocated to specific categories of tasks. These categories were based on the results of formal interview and the knowledge gained by informal discussion and developer induction. The combined results are presented in figure 4.3.

Figure 4.3 Perceived Time Allocation

The largest time allocation was attributed to report writing. It was noted that report writing and experimental design involved a degree of data searching. The results illustrated the group's emphasis on laboratory and field based experimentation. Experimental design and preparation was considered to require more time than the
experimental phase whilst analysis of the experimental data was considered to be the most intensive activity. Ten percent of the group's time was attributed to administrative tasks such as form filling. The remaining time allocation areas were communication with people from both inside and outside of the group and a miscellaneous category which included visits and training. These results provided an indication of relative importance in terms of time allocation and consequent cost to the establishment. The results reflected the group's assertion that information retrieval was an important part of their work. However, the subjective nature of time estimation demanded that these results be considered only as supporting evidence for software development rather than justification in its own right. Consequently, alternative techniques including analysis of the organisation's time sheets and the completion of work logs were implemented as objective measures of group activity.

4.3.6 Time Sheet Analysis

The organisation's official record of work completed took the form of a weekly time allocation log completed by each permanent employee. It provided a record of time charged by each person within the organisation to specific work projects. The records for the three years prior to the development project were retrieved from the central personnel database for analysis.

The time sheet data is combined in figure 4.4 to illustrate the balance of work over the full three year period analysed. This shows a relatively even split between the work attributed to the areas of 'Consultancy', vehicle specific 'Projects' and 'Future systems', including research. The remaining segment is split between 'Administration' activities and 'Other' work including training and teaching time. Omitted from figures 4.3 and 4.4 is the time allocated to annual leave and sick leave.
The particular areas of predicted future effort were discussed with the head of the group who was responsible for managing the levels of effort. This helped to clarify the reasons for the relative allocations between projects and was used as a method of predicting future allocation of effort based on a continuation of observed trends. The resulting prediction of the group's major areas of work for the next three years were as follows:

a) The design and implementation of a mobile concept crew compartment.
b) An investigation of dynamic anthropometry.
c) The development of a mobile battery of workload tests.
d) Individual research work.

The results were also discussed with the other members of the group. This informal discussion highlighted problems with the reporting system. Individuals admitted to the completion of reports months late and confusion between similar cost centres. Consequently, the results as both a measure of past activity and an indicator of future activity were treated with caution.
4.3.7 Voluntary Work Log

Completion of a voluntary work log by the members of the group was adopted as a method of determining the activities of the group in greater detail. Having broadly established the group's use of information for their core work programme, the aim was to reveal specific sources used. Each member of the group was provided with an observation log booklet. They were asked to record all work carried out which required reference to an external source of information such as a reference text or a subject matter expert.

The log booklets were completed over the course of one month prior to the product definition meeting. This time limit was enforced due to both the pressure to move on to product definition, and the need to remove the additional work load placed on the group by log completion. The logged activities are presented below, grouped in information source categories.

1. Experiment preparation (x6)
2. Body size (x4)
3. Workspace design (x4)
4. Vehicle specification (x3)
5. Vision (x2)
6. Administration (x1)
7. Physiological limits (x1)
8. Equipment operation (x1)

The short data collection period limited the results to a snapshot of the group's work allocation. As a result they reflected the current tasking of group members. The importance of information retrieval for experiment preparation was related to an impending field trial. This area of work was expected to tail off following the trial. The other information areas each related to long term work programmes. These areas required consistent effort throughout the working year providing a more representative indication of the group's work programme.

Although the data gathered by this approach was limited, the added detail helped to further define the picture of the group's work programme. The particular areas
highlighted by the observation log included body size, vision and standards relating specifically to military vehicle design. From this point it was possible to construct a list of information subject areas relevant to the work of the group.

4.4 Technical Specification

4.4.1 Platform and Tool Assessment

Figure 4.5 illustrates the linked processes of platform and tool assessment leading to eventual selection of the technical means of meeting the group’s emerging requirements. Whilst the selection of the platform could be broken down into separate issues of cost, performance and user familiarity, the variation in software available for the different platforms prevented their selection without taking account of the necessary software.

![Technical Specification Diagram]

Figure 4.5  Technical Specification

Technical specification was carried out in parallel with requirements analysis, (figure 4.2). As the developer and the group’s understanding of their information requirements became clearer during requirements analysis it became possible to consider commercial software products as a means of supporting the group. However, a review of available software failed to identify any products applicable to the group’s specialised working domain. Instead it raised various concepts for software products which might support the group. This resulted in a gradual shift in the focus of the platform and tool selection process to consider possible development solutions.
4.4.2 Assessment of Platforms

Table 4.4 outlines the assessment of potential platforms to support the project. The central issues against which each platform was assessed were:

1. Cost of purchase and maintenance.
2. Suitability of available prototyping tools.
3. Suitability of available coding tools.
4. User impact.
5. Organisational impact.

Computers larger than the establishment's VAX computer were immediately rejected on the grounds of cost. The VAX platform was considered to be a candidate for development but was rejected due primarily to its limited interface. The group's Sun workstation was rejected due to the complexity of both its operating system and the availability of development tools. The BBC computer was considered as an example of a low end computer platform. However, it was rejected due to its low performance and uncertain commercial support in the future.

The choice of development platform was left to a decision between the two mainstream desktop platforms of the IBM compatible Personal Computer (PC) or the Apple Macintosh. These two platforms were similar in terms of cost and performance. Although the PC platform was the most widespread desktop platform within the establishment, numbers of each were small. The low level of computer usage by group members was reflected by the absence of an organisational computer policy. This allowed freedom of choice between these two competing platforms for development and implementation. The PC held a larger market share which was reflected in the range of commercial software available for it. However, in the area of software design and development both platforms were well supported.
<table>
<thead>
<tr>
<th>Platform</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mainframe Computer *</td>
<td>Peak Performance</td>
<td>High Cost</td>
</tr>
<tr>
<td></td>
<td></td>
<td>High Maintenance</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Professional Tools Only</td>
</tr>
<tr>
<td>VAX Microcomputer *</td>
<td>High Performance</td>
<td>Text Interface</td>
</tr>
<tr>
<td></td>
<td>Existing Network</td>
<td>Centralised Control</td>
</tr>
<tr>
<td></td>
<td>Large Storage Space</td>
<td>Professional Tools Only</td>
</tr>
<tr>
<td>Sun Workstation *</td>
<td>High Performance</td>
<td>Specialised Operating System</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Professional Tools Only</td>
</tr>
<tr>
<td>DOS Desktop Computer (PC)</td>
<td>Low Cost</td>
<td>Standard Performance</td>
</tr>
<tr>
<td></td>
<td>Graphical Interface</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Broad Range of Off-the-shelf Software</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Broad Range of Tools</td>
<td></td>
</tr>
<tr>
<td>Macintosh Desktop Computer</td>
<td>Low Cost</td>
<td>Standard Performance</td>
</tr>
<tr>
<td></td>
<td>Graphical Interface</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Simple to Network</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Broad Range of Off-the-shelf Software</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Broad Range of Tools</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Graphical Operating System</td>
<td></td>
</tr>
<tr>
<td>BBC Computer *</td>
<td>Very Low Cost</td>
<td>Low Performance</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Basic Development Tools</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Basic Interface</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Little Commercial Support</td>
</tr>
</tbody>
</table>

* Rejected

Table 4.4 Assessment of Computer Platforms

4.4.3 Assessment of Development Tools

The similarity of the remaining platforms led to selection on the basis of available programming and prototyping tools. Consideration of professional programming tools for the two platforms showed many similarities. A full range of professional development languages was available on both platforms. The functionality and complexity of these professional tools was similar.
A greater distinction between platforms became apparent in the evaluation of high level programming/prototyping tools. The tools considered for the PC included ‘ToolBook’ (NEOW Ltd. 1990) and ‘Omnis 5’ (Blyth Software Ltd. 1991). Both applications supported the rapid application development in comparison with the conventional programming approaches. Both were examples of a high level approach in which the programming code was encapsulated in objects which could be arranged to form associations and complete limited functions. Both suffered from a comparative reduction in operation speed associated with this approach.

The equivalent programming/prototyping tools considered for the Apple Macintosh computer were HyperCard (Apple Computer 1989) and SuperCard (Silicon Beach 1989). HyperCard’s relative simplicity and object oriented approach to application development made it the prime candidate for selection as the project’s prototyping tool. Its major limitations were the restricted screen size and the lack of support for colour. SuperCard closely matched the HyperCard structure and offered a superset of the HyperTalk language (Young et. al. 1990), whilst overcoming these reported limitations. Martin (1994) considered SuperCard to be the most suitable application for non-programmers to develop serious applications by providing real access to the interface elements of the computer.

At first, SuperCard was only considered as a prototyping tool to support the Design phase of the proposed lifecycle. However, in addition to its prototyping capabilities, SuperCard supported the conversion of its prototypes into standalone applications. This suggested the possibility of developing a deliverable product using this tool. This offered the opportunity of extending the prototyping phase to include acceptance and delivery without the need to use another tool. This could remove the delay associated with the translation of the prototype specification into application code, streamlining the development lifecycle. If effective, it also removed the need to allocate developer resources to learning a second language in order to code the product. This represented a major deviation from the initially proposed development lifecycle, (figure 4.1). The success of this approach relied on the ability of the tool to both prototype and deliver a product plus the ability of the modified lifecycle to guide effective application development.
4.4.4 Provisional Platform and Tool Selection

The advantages offered by SuperCard as a means of supporting prototype design, development and implementation led to its provisional selection. The Apple Macintosh computer platform was chosen by association. The equivalent high level development tools on the PC were considered to offer less scope for both prototyping and product delivery. This view was reflected by large number of HyperCard/SuperCard applications reported in the literature (Johnson and Johnson 1988, Nicolson 1990, Young et al. 1990), in comparison with alternative high level development tools. Professional programming tools were rejected in favour of this high level prototyping tool due to SuperCard’s comparative ease of use. The promise of a shorter learning period (Neilsen et al. 1991) and the rapid development of graphical applications presented an important advantage over the other available development environments.

The provisional selection of SuperCard presented uncertainties concerning the limitations of the SuperTalk language and the comparative execution rate of high-level applications. Consequently, final acceptance of SuperCard as the primary development tool followed the development of test applications, (chapter 5). The development of these applications tested the ability of the chosen software and hardware to support the central functions considered necessary to handle the information identified by the parallel analysis of requirements.

4.5 Product Definition

4.5.2 Formal Group Discussion

The proposed approach to specification was the parallel activities of requirements analysis and technical specification, culminating in a specification of the product as a guide to the continuing project, (figure 4.2). The techniques applied to the analysis of requirements confirmed the importance of information reference for the group’s work. They succeeded in categorising the work areas of the group whilst failing to highlight an overriding candidate for immediate support. However, they provided sufficient information to exclude commercial products. The assessment of platforms and tools highlighted a computer platform and development tool package which promised to
support tailored software development within the resources of the project. The final process of the specification phase aimed to combine the findings of these two processes in a form that would allow the project to advance to the next stage of the development cycle. Continuing the project's user centred philosophy, the approach chosen for product definition was formal group discussion.

The picture of the group's work programme and information requirements built up from the previous analysis procedures supported the construction of a list of information areas relevant to the work of the group. The list was distributed to all members of the group for comment. An amended version was distributed at the start of the meeting. This list of possible information areas provided the focus of the meeting and acted as a loose agenda. Whilst the technical options were presented to the group, the decision for tool selection was retained by the developer with the agreement of the head of the group. The aim of the discussion was to define the continued work programme of the project by selecting an information area for support.

The meeting was chaired by the developer to avoid any single member of the group dominating the discussion. Whilst all members were involved in the discussion, their relative contribution to the meeting varied. The individuals who had the greatest influence on this final specification activity were either senior in the group hierarchy, or experienced in the use of computers. By chairing the meeting the developer was able to involve the quieter members but could not artificially balance the relative levels of input.

The meeting started by eliminating information areas unanimously considered to be inappropriate for further consideration. Reasons for rejection included low requirement, inappropriate subject area or lack of available data in the subject area. The remaining areas were then compared against the criteria of:

a. Its relevance to the overall work programme.
b. Its suitability for computer based support.
c. The potential improvement over the current source of information.

The result of this process was an agreed priority list of information areas considered appropriate for subsequent development. The priority list is presented in table 4.5.
The meeting provided a forum for group members to voice their opinions concerning the form and content of the future system. The specific comments raised were:

- Only include information relating to vehicle design and specification.
- The list indicates the proposed order of module development.
- The priority list should be re-evaluated throughout system development.
- Many category areas included are interconnected.
- Source documentation should be presented with all information.
- Data should be taken where possible from existing databases.

Body size was selected as the highest priority area for computer support. Accurate, vehicle related, anthropometric data was agreed to be an important requirement for both project specific work programmes and occasional consultancy. 'Body Size' compared favourably with the less specific information area of 'Workspace Design'. Whilst 'Workspace Design' references were available to the group in paper form, they were unable to access the anthropometric database which was held in its raw form on...
the establishment's VAX mini computer. Instead, the group relied on a less population specific military standard for anthropometry data.

The prioritised list of information areas was adopted as the work programme for the developer for the purpose of project management. The selection of anthropometry as the first project area allowed the developer to concentrate on providing a tool for retrieving and presenting the establishment's database.

4.6 Discussion of Methods

The traditional lifecycle, from which the specification phase was taken (figure 4.1), would normally start from a clear concept which would then be formally described as the basis of design and coding. The socio-technical approach adopted from the start of this project placed different requirements on the specification process. It became necessary for the specification phase to consider user requirements and technical solutions from the loose objective of supporting the work programme rather than a specific goal such as detailing a firm product concept. The project adopted user centred techniques for requirements analysis in parallel with the developer's evaluation of the technical options in relative isolation.

Requirements analysis for the VDS group identified three sources of information relating to the requirements of the group and its work programme. The sources and the techniques applied to their investigation are reported in table 4.6.

<table>
<thead>
<tr>
<th>Source</th>
<th>Applied Technique</th>
</tr>
</thead>
<tbody>
<tr>
<td>The users</td>
<td>• Semi-structured interview</td>
</tr>
<tr>
<td></td>
<td>• Informal discussion</td>
</tr>
<tr>
<td>The environment</td>
<td>• User work logs</td>
</tr>
<tr>
<td></td>
<td>• Developer induction</td>
</tr>
<tr>
<td>The past</td>
<td>• Analysis of organisational records</td>
</tr>
</tbody>
</table>

Table 4.6 Sources for Requirements Analysis
The project's approach for requirements analysis supported the final product definition phase in different ways. The formal techniques ensured even consultation across the group and highlighted candidates for development in the form of information areas. At the same time the informal approach of developer induction and free discussion improved the understanding of all involved about the group's requirements, their relative roles in the project and how to communicate their views. All these features were important in setting up an informed forum in which all members of the group were able to contribute to the determination of the project's work programme. The judgement on whether the conclusions of this specification process were correct depends on a longer term view of the development project's success, (chapter 10). However, the short term necessity of providing a starting point for development based on the informed consideration of the users' requirements was achieved.

The following sections consider the suitability of the techniques applied to the project's specification phase. Conclusions relating to the conduct of specification within the context of the project's environment are presented and the amendments made to the project's proposed lifecycle are illustrated.

4.6.1 User Interview

Interviews with each member of the user group were found to provide an effective starting point for the process of user requirement specification. They introduced the developer to all members of the group, opening lines of communication through which informal discussion continued throughout the rest of the project.

Whilst questionnaires may be highly suitable for determining the views of a large user group, the interview was found to be preferable for the project's small number of users. In addition to providing consistency between subjects, it offered the developer the flexibility to expand the discussion beyond the set questions. This was important at the start of the project allowing both the developer and the user to explore the issues surrounding the project. In contrast, distribution of a questionnaire requires the author to anticipate all relevant issues from the start.
Problems encountered with the interview process centred primarily on mutual availability. However, these problems were diminished by the benefits of the technique. As a starting point it established contact between the developer and the user group which went on to provide continual, if unstructured, feedback throughout the project. It demonstrated the project's user centred approach establishing each user's stake in the project. This helped to encourage the group members' support and participation at this early stage.

An alternative to the approach of individual interviews would have been the use of a further group session during which the issues raised could have been addressed by the group as a whole. This approach was rejected due to both the difficulty of arranging a common time to meet and the risk of losing the input of less forthcoming individuals.

4.6.2 User Log

The response to the voluntary task was low. A total of 22 work reports were received from the group over the period of one month. The information was further unbalanced with a disproportionate level of response from the junior members of the group. Suggested reasons for the poor response include the following:

- High time pressures of work.
- Low motivation to complete the log.
- Individuals working in areas not requiring reference material.

The factors listed appeared to increase with seniority within the group. The level of responsibility and the associated time pressures grew across the hierarchy of the group. This required these individuals to practice tight time management. Completion of the voluntary work log had to compete with a large range of other demands on the individual's time. Discussion with the senior group members revealed that an additional reason for their low response was their current work in familiar subject areas. As their experience grew they were more likely to be working within an area in which they were an expert.

Despite the low response level, this technique provided valuable detail concerning the use of information within the group. Whilst compulsory completion of the work log
would have increased the number of returned work logs, this approach would have risked the alienation of the users and been difficult to enforce for the more senior members of the group. User participation on this footing would have been unsustainable. It was essential that the motivation for user input was rooted in a desire to achieve a suitable product.

Encouragement to complete the logs was regularly applied when collecting the forms at weekly intervals. More frequent collection might have increased the response level. However, without prolonged data gathering this technique remains limited to a snapshot of the group's true work programme. The reliability of the results must be considered in this light. As a consequence, the level of effort expended on this activity must retain a reasonable proportion to the value of the information it provides. The balance struck in this case was considered to be appropriate.

4.6.3 Historical Data

Analysis of the organisation's time sheets attempted to apply existing information sources to the requirements of specification. This was seen as a convenient source of objective information from which to build a model of the group's work programme. However the broad nature of the organisation's costing categories, in conjunction with the inaccuracy within the recording system, limit the value of the results. Although they provide an rough indication of the group's past work programme, they could not be relied upon as a firm guide of the future work of the group. Whilst further defining the picture of the group and its work, it's influence on the eventual selection of support areas for the proposed information system was minor. In this case the balance between expended effort and the information gained was considered to be inappropriate. A more thorough consideration of the organisation's own use of this information may have revealed its broad nature and relative unreliability thus flagging it as an inappropriate source for the VDS project.

It is notable that the new management procedures introduced during APRE's transition to CHS included a new time accounting approach in which each individual's time allocation was collected weekly and in greater detail. This updated approach would have provided more relevant information. The use of existing organisational data may provide
useful data for analysis of requirements. However, the form of the data normally remains beyond the control of the development project.

The time sheet data was not the only organisational information source used in the project. An internally maintained telephone enquiry file was used in the design of the Standards module, (section 7.6.1.2). However, other organisational sources such as the library enquiry file provided little relevant information. The work involved in gaining access and analysing these sources was often out of proportion with the amount of relevant information found. Whilst the library enquiry file showed a relatively low access rate, experience of the group's working practice showed that other sources, such as internally held documents and other experts, combined to increase the overall levels of document referencing.

4.6.4 Informal Discussion

Where the available time for user/developer contact is limited, information must be gathered efficiently in an accountable manner. Formal methods provide a controlled approach to gathering this information. Whilst the formal methods applied to this project attempted to do this, user/developer communication provided by the in-house environment provided a vital supplement this source.

The formal techniques applied to the analysis of requirements provided consistency and progress markers for the project. However, these techniques demanded time and effort from both the developer and the users due to the inherent planning, analysis and reporting involved. They required considerable planning to ensure that all relevant issues were addressed. Whilst the information exchanged by informal discussion required effort on behalf of both groups, it was able to fit around the commitments of both parties. This removed much of the inconvenience encountered by both sides in applying formal techniques. However, it relied on the willingness of the group members to become involved in the discussion and risked a disproportionate influence on the project.

Informal discussions were not recorded as this was considered to be both impractical and inappropriate. Such 'formalisation' would have changed the nature of the activity, potentially preventing the unguarded exchange of thoughts between the developer and the
users. Consequently, the impact of this source of information on the development process cannot be quantified. However, observations could be made concerning the role of informal communication on the specification process.

Discussion prompted the user group to consider:

- Potential forms of a computer based support tool.
- What information might be supported.
- Their expectations for an 'in-house' software application.

Discussion prompted the developer to consider:

- Organisational structure and its lines of communication.
- The structure and relationships within the user group.
- The specific characteristics of the group members.
- Existing problems with the group's work programme.
- The strengths and weaknesses of the user group.

An important feature of this process was the exchange of information in both directions. In comparison, the formal approaches concentrated on retrieving information from the subject. Although the flexible approach to the interview allowed the user to ask questions, the emphasis was on the user as the information source. This was reflected by the continuation of discussion relating to the project beyond the time set aside for formal information gathering. Topics of discussion ranged from aspects of the group's work programme to the project's scope for software development.

The high level of informal discussion reduced the impact of the formal specification procedures. In a lifecycle where these procedures provide the user's only contact with the developer their success is critical both in defining the product and in forming the user's response to the project. In the VDS project the main effect of the formal user interactions was to provide project milestones. They were unable to provide a definitive product specification. Even after the formal group discussion, continued discussion regularly reassessed the project's direction and the form the products should take.
An alternative approach to providing a more accountable approach to the discussion process might be the regular group sessions to discuss all aspects of their work, including the development project. This would allow user views to be recorded throughout the project. However, the problems encountered in achieving a full group meeting experienced in the final product specification activity of this phase suggested that a regular group meeting would not be achievable within the group's working environment.

4.6.5 Developer Induction

The developer's integration with the user group relates to the approach of the designer as an apprentice (Beyer and Holtzblatt 1995). They report the benefits of developing an intimate relationship between the designer and the software customer. They observed that seeing the work helps the investigator understand the structure and detail of work practice, in turn highlighting the key features of their daily activities. These findings match the experiences of the VDS project. By developing this close working relationship potential barriers to information transfer were removed. The members of the group were always prepared to explain and demonstrate all aspects of their work. Actual completion of these tasks by the developer, whilst maintaining a separate perspective, developed a depth of knowledge which would not have been possible from the traditional requirements gathering techniques alone.

Whilst comparing the VDS project with the cases reported by Beyer and Holtzblatt, the developer's integration within the group went beyond the concept of the developer as an apprentice. Whilst an organisational separation remained between the developer and the VDS group, the distinction diminished as the project progressed. As a consequence, the project could be considered to be a case of internal development by the user group rather than development by an apprenticed designer.

The time cost associated with the reallocation of developer effort was recognised to reduce the direct effort on the development project. This was considered to be both unavoidable and offset by the advantages of the induction process. The requirement to pool effort was a consequence of group size and management structure. This pooling remained a constant feature throughout the period of the development project. The move towards centrally managed human resource pools did not fully take effect until the end of
the project. Involvement in the broad work programme of the group provided the developer with:

a. Knowledge of the group's work specific information requirements.
b. Knowledge of the organisation's working procedures.
c. Developer/User communication at an informal level.
d. Reduction in the total time available for specific system development.
e. Interruption of project development sequences.

In addition to these developer specific issues, developer integration transformed the project from a remote process into an activity in which each group member had a role and a stake in its success. This contributed to their support and encouragement by the user group. In contrast, it is likely that development by a remote team or individual would have may have met a degree of caution and possibly resistance within the group.

4.6.6 Technical Specification

The task of platform and tool selection was more straightforward than the analysis of user requirements because the issues relating to selection could be clearly defined. The approach taken was to identify the technical options, reduce them to a short-list and to compare remaining options against set criteria. This approach, whilst simplistic, was effective in selecting a computer and associated tools for the development of the future system. The technical assessment could have been formalised by documenting the comparison within a matrix. A matrix can provide a structure which can help to ensure that the developer considers the necessary features of a proposed system. An example is provided by the functionality matrix which forms part of the HUFIT toolset (HUSAT 1989). This matrix requires the developer to compare task and user characteristics against hardware and software functionality. It was considered to be inappropriate at this stage of the project as the concept product was not sufficiently defined to specify the software functionality demanded by the matrix. However, the HUFIT toolset was returned to later in the project as a guide to the more detailed design of one of the project's software products (chapter 8).
The best indication of the suitability of the simple candidate comparison for technical specification rested in the subsequent ability of the chosen system to support the work of the group. The chosen platform went on to form the basis of the group's regular work programme. By the end of the project, the group had acquired desktop Macintosh computers for each member. These were used for both commercial software and the access of the developed information system. An internal network was set up as part of the project which provided each member of the group with access to the tailored software and a central printing and scanning facility. Whilst the success of the chosen platform does not allow the conclusion that other platforms would have been less successful, it does confirm that the selection process considered the features necessary to identify an appropriate solution. This conclusion was further supported by the organisation's subsequent decommissioning of the VAX system which was rejected by the comparison process.

Whilst the head of the group was ultimately responsible for the acquisition of the project's technical tools, the developer retained control over the comparison and selection process. The rest of the group showed little interest in the technical options preferring to provide input in areas directly relevant to their work. The extra effort which would have been required to formally involve the entire user group in the technical evaluation process could not be justified as they were neither inclined nor in a position to become directly involved in the technical evaluation process.

4.6.7 Formal Product Specification

The project's approach to formal product specification was through a group session to decide the continuing work programme of the development project. An increasing recognition of the importance of group session approaches to requirements capture and analysis have led to a range of approaches including Joint Application Design, Quality Function Deployment and Cooperative Requirements Capture (Macaulay 1996). These approaches formalise requirements capture ensuring the involvement of stakeholder group's and facilitating communication. As noted for the formal structured development methodologies, (section 3.5.1), these approaches targeted large scale development projects. Whilst the benefits of a group session were recognised, adoption of an approach designed for large scale development was not considered to be appropriate. Instead the study sought to apply the principles of effective group session
implementation for a single meeting to establish a priority list for the project's development programme.

The formal group meeting was an extension of the informal discussion process from which the participants had developed a view of the project and its necessary direction. As these informal discussions did not involve all members of the group to the same degree it was important to seek a balance of views at a formal meeting of the project's stakeholders. This ensured that the views of all users were considered in the eventual selection of information areas to support. At the same time, it provided a milestone for the development project.

Despite the many different views within the group, the meeting was able to arrive at an immediate work programme. The approach of list prioritisation avoided the potential conflict associated with outright rejection of competing concepts in favour of a single 'winner'. A group member arguing for a concept rejected in favour of a majority view might be disinclined to support the subsequent development project. This risk appeared to be reduced by retaining minority concepts on a priority list for possible future development.

The involvement of the entire user population in this decision making process made all of the users part of a team in which the developer was simply the means of implementing the decisions of the group. This approach of user ownership and responsibility for the system encouraged the continued involvement of the group members throughout the product lifecycle.

The result of the specification phase's formal product definition fell short of the more detailed definition anticipated at the start of the specification phase. The investigation of the group's activities did not suggest a single product which might serve the majority of the group's information requirements but rather a wide range of potential products, each of which might contribute to the work of the group. The result was a priority list of products for development following product definition. Product definition in this form did not attempt to suggest the functionality required to support the proposed information areas. Rather the product of the project's specification phase resembles the minimum critical specification described by Cherns (1976).
Eason (1988) rejected the notion of achieving a 'right first time', detailed specification preferring instead flexibility in specification to accommodate changing requirements. The project's prioritised list was minimal in its detail of the required product. At the same time, it set out a considered view of the user's information requirements arrived at through user consensus and based on user centred techniques for requirements analysis. The project could move forward with this top level view to design tools to support the identified requirements whilst retaining the flexibility to change the development plan to accommodate changes over time.

4.6.8 Lifecycle Iteration

An important feature of the user group relating to lifecycle design emerged during this early stage of requirements specification. This was the group's generally low level of computer software experience. As a consequence, the users lacked a model on which to base a concept of how a system could be tailored to their requirements. They required guidance in the choice of suitable areas for software support as well as the form that such a system might take. It quickly became apparent that without this guidance the users were unable to specify their requirements of the system in sufficient detail to proceed to the traditional design phase of the waterfall lifecycle.

The narrow working area of the group reduced the suitability of other available software systems as models for the proposed VDS system. The specification sequence documented in this chapter was able to broadly define the information areas suitable for development. However, it could not provide the detailed specification of information content and functionality required for software development to begin. In order for the project to move forward it was necessary to explore the user requirements further based on the information already determined. The method proposed was concept testing through rapid prototype development. The prototype would then serve as the model against which the specification of the full system could be detailed.

The lifecycle framework was revised, replacing the traditional design phase with a rapid prototyping phase, (figure 4.6). The new model accommodates the user's inability to firmly specify their requirements for the system. It aimed to further define the users' broad specification through rapid prototype development and testing. The loose specification of suitable information areas to support was considered to provide a
suitable starting point from which prototype development could be used to explore possible solutions to supporting each information area. A semi-functional prototype of a product concept could then be constructed and presented to the user population as the model for more detailed specification.

Figure 4.6 Lifecycle Model for Design Exploration

The lifecycle, (figure 4.6), recognised the limitations of the specification process, deferring the specification of product details to a design phase in which alternatives should be explored through rapid prototyping.

Figure 4.7 Provisional Lifecycle Model
The project's provisional selection of a software prototyping tool which promised the capability of delivering a completed product was incorporated in the model with the removal of the coding phase of the lifecycle, (figure 4.7). Acceptance of this shortened lifecycle for the development of the VDS system required confirmation of SuperCard's ability to deliver a fully functional information system. Chapter 5 presents the testing of this issue and incorporates the results in a further revision of the lifecycle model prior to the development of the VDS system.

4.7 Conclusions

The project's specification phase investigated the group's requirements and identified an approach to supporting these requirements. The project's development priority list defined the project's immediate work programme. Conclusions made at this point in the project are presented below.

1. Induction provided the developer with a practical illustration of the user's work programme and formed a channel for communication. The cost of this approach was the diversion of development effort later in the project.
2. Structured interview provided the flexibility to expand discussion beyond the issues anticipated at the start of the project.
3. Informal discussion suited the in-house environment, providing a two-way transfer of information between the developer and the users.
4. Lack of control over the organisation's time allocation records made this source of information unreliable.
5. Voluntary completion of a work log provided relevant information for the specification process. However, the short collection period and varied completion biased the results.
6. The absence of a clear user concept of a required product prevented detailed specification.
7. Tool selection was based on available finance and ease of use. Platform selection was linked to the chosen development tool.
8. Despite the wide variety of information sources analysed, the view provided was judged to be equivalent to a minimum critical specification.
5 Test Application Development

5.1 Chapter Overview

The selection of a computer platform and a software development tool relied on a broad evaluation of the available technical options. The chosen tool's ability to support the project in terms of functionality and scope was explored in the development of three test applications prior to the development of the Database Application for Vehicle Ergonomics (DAVE) system software. Chapter five considers the development of these applications and discusses the design of an iterative prototyping cycle to match the strengths and limitations of the development tool and the environment.

The structure and functionality of the SuperCard development tool is presented. Acceptance of the tool was based on confirming the functional scope of the test products. Three functional areas were identified as necessary to support information system development. Demonstration of product functionality in the areas of text handling, numerical data processing and graphical manipulation was proposed. The chapter describes the simplistic model of iterative design adopted as an initial guide to the design and development of the test applications. The development of each application is reviewed, and their separate aims are explained.

The chapter considers the design of the DAVE system structure prior to module development. It also reports the evolution of the DAVE interface standard from the code and graphics of the final test application. The chapter continues by assessing the contribution of each application to the overall development project, in relation to the associated costs.

The chapter presents a development process model based on the experience gained from the development of the test applications. This model matches the structure and flexibility of the development tool. The model's iterative cycle suggests the initial design and iteration of a non-functional prototype followed by the addition of functions. This approach to design exploration was considered to reduce the risk of lost effort over the early design cycles, during which most changes were made to the test applications.
5.2 Acceptance of the Development Tool

5.2.1 Tool Investigation

Chapter 4 presented the process adopted for the project’s specification phase. Requirements analysis provided a minimal specification tied to a proposed technical solution. The technical specification activity, as represented in figure 4.2, identified a development tool and platform. This provided an indication of what was achievable prior to the final product specification activity. Following the technical specification of SuperCard as the preferred development tool, the functionality and scope of the tool was explored to confirm or reject the decision.

The approach taken to confirming the scope of the tool was through the development of test applications to investigate the limits of SuperCard as a development tool. Failure of the tool to support these functions would lead to re-evaluation of alternative development tools prior to the formal development programme.

The functions considered necessary for the construction of a group specific information system were categorised as text, numerical and graphic handling. Separate applications were planned to explore SuperCard's functional scope in each of these areas. Acceptance of SuperCard as the project’s primary development tool depended on confirmation of its ability to support the required range of functions. This activity continued in parallel with the completion of the requirements analysis activity. With the exception of the final test application, they were largely completed prior to the formal group meeting to specify the product of the development project.

5.2.2 SuperCard

The provisional selection of SuperCard as the project development tool was based on a combination of the tool's ease of use, promised programming scope, and potential for product delivery, (section 4.4.3). Its structure and functionality were important in defining the approach to development using the tool. At the same time the developer was able to take different strategies towards a single programming goal.
The structure of a SuperCard project is illustrated in figure 5.1. The main components of a SuperCard project were windows, cards, backgrounds and objects. The project's windows could contain a series of cards. Each card had an associated background which could be individual or shared with other cards. Objects such as buttons, fields or graphics could be positioned on each card or background. When viewed, a card's objects were overlaid on the associated background.

Any of the listed components could contain SuperTalk code. SuperTalk is a high level language which allowed each component to respond to interaction, process information, pass messages to other components, etc. SuperTalk provided full interface control including sound and animation. External code routines created using professional development tools could be used to extend the functions available within SuperTalk to support text filtering, video playback, etc.

An important feature of the SuperCard environment was the inclusion of a script editing utility, absent from HyperCard at the time of selection. SuperEdit allowed application elements to be tested and altered as soon as the code was written. This gave the programmer the ability to immediately assess the consequences of a design decision by

Figure 5.1 The Structure of a SuperCard Project
running the application and tracing the results. Errors revealed could be diagnosed and corrected by stepping through the code one line at a time.

The advantages of simplified development were balanced by known limitations to this approach to programming. By directly controlling the detailed computer processes, lower level languages such as 'C' (Symantec 1989) provided control over every aspect of the programme. The abstraction of the SuperTalk language constrained the programmer to linking standard routines. Whilst reducing the degree of control, this approach simplified the development process and made the code more readable. The drawback was the time penalty for translating the language to machine instructions in order to complete each function. The use of a compiler, Compilelt (Heizer Software 1990) was considered for improving execution rate by translating the language before saving it as an application. However, the compiler was rejected as it failed to achieve an overall reduction in execution time for the applications compiled.

5.2.3 A Development Process Model

The primary role of test application development was to confirm SuperCard's ability to support the necessary functions of an information system. In addition, it provided an opportunity to determine a process model for the development of the proposed information system. The model needed to support the characteristics of the development scenario, such as close user involvement, as well as the strengths and weaknesses of the development tool.

Requirements analysis revealed the users' need for support to define their product concept beyond simply a specific area of information. Whilst it had become possible to build a good picture of their current work practices, the leap to specifying a product was blocked by many uncertainties. Some of these uncertainties were described by Harker et. al. (1992) in their discussion of the ways in which software requirements change. They noted that requirements were likely to emerge from user participation in the specification process and go on to evolve further as a consequence of the tool's effect on the work process. They argued for an acceptance of change in requirements as a normal part of software development, to be accommodated through more appropriate user involvement and an evolutionary approach to development. Hekmatpour and Ince (1987) suggested evolutionary prototyping as an effective method for responding to
changing requirements. Rather than restricting prototyping to a set design phase, (figure 3.4), evolutionary prototyping attempts to retain flexibility in design throughout the development process.

The project needed to move forward from the group’s minimal specification of appropriate information areas. The provisional selection of a high-level tool for development presented the opportunity to apply this approach to both exploring software solutions and modifying chosen solutions to accommodate the predicted drift of requirements. Prototyping provides a method of presenting, testing and iterating design concepts through an accessible medium for developer/user communication. This iterative cycle is presented in figure 5.2.

![Prototyping Model](image)

Figure 5.2  Prototyping Model

Section 3.6.3.1 distinguished between alternative forms of prototyping. The main difference between the reviewed techniques was the eventual fate of the prototype. Rapid prototyping discards the prototype for a fresh design or the transfer to formal coding (Bickford 1994). The intention of bypassing the coding phase using SuperCard pointed towards evolutionary prototyping in which the prototype evolves into the final product.

A necessary feature of this prototyping cycle is the involvement of the user in the process of design evaluation (Gomaa 1983). This approach was considered to suit the development project due to the daily contact between the developer and the users.

The test application development period provided an opportunity to investigate the strengths and weaknesses of the development tools and environment. These characteristics could then be applied to define a more detailed software process model to match the characteristics of the tool and the development environment. Assuming
selection of the proposed tool, this model would then form the basis of a development phase for the lifecycle of the main project.

5.3 Test Application Development

5.3.1 Overview

Following the selection of SuperCard as the development tool, it was used to construct small evaluation applications. Three applications were rapidly developed exploring the abilities of the tool to handle the following areas:

- Text handling
- Numerical data handling
- Interactive graphical interface support

The project purpose of this test development period was to:

- Determine functional limitations of the programming language.
- Provide experience in applying the tool.
- Gauge the software development time-scale.

In addition to confirming the functional scope of the software, the test applications provided the opportunity to ensure that the performance decrement associated with the high level code did not compromise usability. Software response to user interaction was considered to be an important issue in achieving an acceptable application. Target performance levels were not set as it was recognised that these were task specific. For example, users were prepared to wait for the computer to ‘boot-up’ but expected immediate feedback from keyboard interaction. The test application development process allowed user response to any delays associated with data handling functions to be gauged as a guide to acceptability.

Although the test applications were not developed to meet specific user requirements they were each designed to support a facet of either the development project or the group’s regular work programme. The Text application was used to index references gathered in relation to the development project. The Numerical application was used to
separate and display the acquired Anthropometric data. The Interface application was applied to the group's regular work area of control panel assessment.

5.3.2 Text Handling

The first of the test applications to be developed was the Text application. The functions considered central to the use of text based data by a future information system were set out by the developer, (table 5.1).

<table>
<thead>
<tr>
<th>Required Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Read from text files</td>
</tr>
<tr>
<td>2 Write to text files</td>
</tr>
<tr>
<td>3 Search text documents</td>
</tr>
<tr>
<td>4 Sort text lists</td>
</tr>
<tr>
<td>5 Display text documents</td>
</tr>
<tr>
<td>6 Support input of text</td>
</tr>
<tr>
<td>7 Support user editing of text</td>
</tr>
<tr>
<td>8 Print text documents</td>
</tr>
</tbody>
</table>

Table 5.1 Functional Specification for the Text Application

The listed functions were added to the emerging prototype and refined. The prototyping environment allowed elements to be added and deleted, resized and repositioned. In order to assess the application's ability to handle large document sources it was applied to the indexing of project related abstracts, (figure 5.3).

The development of an abstracts database demonstrated the ability of a SuperCard application to store and access a limited text database (<300 references). Text searching and retrieval rates suggested the possible support of much larger databases. As the database grew, the advantages of text searches began to outweigh paper based, page searching approach.
Paper versus Computer Implementations as Mockup Scenarios for Heuristic Evaluation

Jakob Nielsen

A taxonomy of the various forms of scenarios in the user interface field is given, including a discussion of different forms of mockups. A single interface design for a video-tape system was implemented as a mockup in two different mediums. As a paper mockup and as a running prototype using HyperCard. These two versions of the same design were then subjected to heuristic evaluation by two similar groups of evaluators. Both versions contained the same fifty usability problems, but there were great differences in the types of problems found by the two groups of evaluators. This indicates that the medium in which a design is presented will have a major impact on what kind of usability problems can be discovered using heuristic evaluation.

Technical University of Denmark

Figure 5.3 Text Application

The time cost of text input was found to be the limiting factor in construction of the retrieval tool. The construction of the main application including the full functionality specified by the list was completed within two weeks. In comparison, building up the document database represented a labour intensive task which involved occasional input over the period of a couple of months. A combination of direct entry and optical character recognition (OCR) was used to generate the database. The advantages of OCR prompted the acquisition of a page feeding scanner, later in the project.

5.3.3 Numerical Data Processing

The second test application investigated the development tool's ability to handle numerical data. The parallel requirements analysis highlighted the establishment's anthropometric database as a potential area of support for a future software tool. This was later confirmed by the formal group meeting. This data source was chosen to test SuperCard's ability to handle numerical data. The functions considered important by the developer were laid out in a list, (table 5.2).
Table 5.2 Functional Specification for the Numerical Application

The purpose of the numerical test application was more defined than the Text application. Rather than simply confirming SuperCard's functional scope, this application was used to sort and check the establishment's anthropometric database. This allowed the data to be considered as a possible information source for the group. The unformatted data existed as ASCII files on the establishment's VAX computer. The use of SuperCard to develop a data separation application involved the main numerical handling functions considered necessary for a future information system. The application was required to read the VAX file, separate the data, test the separated data for accuracy and output the data to a separate file.

Figure 5.4 shows the plots of the separated stature measurements from four populations of anthropometric data. Development of the Numerical application demonstrated a similar balance to the Text application in terms of development time. Construction of the functional plotting application was completed within a week in comparison with the three weeks taken to gather and decode the compressed anthropometric database. Construction of the application demonstrated adequate numerical support within the SuperTalk language to undertake the data manipulation required for the retrieval and presentation of the establishment's anthropometric database.
5.3.4 Interface Support

Development of the Interface test application grew from a small utility constructed in support of the group's regular work programme. Whilst the previous test applications confirmed SuperCard's ability to support the central functions of data support and presentation, this final test application explored its control over the user interface. This led to the design of the interface style adopted for the DAVE system.

The SuperCard utility arose from a group project to evaluate a hardware panel based on a design 'blue-print'. The group normally used paper based prototyping to help to visualise the task sequences linked to the operation of a proposed panel for past evaluations. The new prototyping capability offered by the continuing assessment of the SuperCard tool presented the possibility of developing a functional representation of the proposed panel design.

The panel drawing was scanned. SuperCard buttons were then overlaid on this image. Each button was instructed to respond to interaction with a representative sound, basic
animation and initiation of interface features such as warning signals. The computer representation of the interface was presented at a 1:1 scale on a touch screen. This interactive model was then used to conduct a walk-through evaluation of the panel design leading to design recommendations. The simulation highlighted features relating to feedback and functional grouping of interface components. This practical demonstration of the design limitations assisted communication with the designers of the hardware panel.

The continued development of the panel prototyping concept as an Interface test application followed the success of this initial utility and the suggestion of enhancements to make the tool more widely applicable. The suggested improvements were:

- Added realism.
- Smooth animation for button feedback.
- Simple repositioning of the panel components.

The result was the evolution of the 'interactive blue-print' into the reconfigurable Hardware Panel Prototyping application, (figure 5.5). This interface added sound and animation to a panel of scanned images of real hardware panel components. The use of this approach required the selection of an additional development tool to manipulate scanned images. The selected tool was the industry standard image manipulation package, Photoshop (Adobe 1989). The advanced graphic manipulation facilities provided by the tool, plus its ease of use, ensured Photoshop's acceptance as a development support tool.

The interface test application demonstrated the strength of SuperCard for constructing graphical user interfaces. It was able to simply handle multimedia components such as graphics, sound and animation to produce an advanced interface. The incorporation of interface components such as speech input/output and video presentation were not tested as they were seen to fall outside of the direct requirements of the proposed project work programme. Both video and voice communication were supported in a later release of the development tool but remained outside the determined requirements of the group's information system.
The Interface test application revealed an important limitation of the development tool. It became apparent when using the Interface application to construct a detailed panel that the simulated controls responded more slowly as the number of other interface elements was increased. Subsequent module designs overcame this limitation by including most of the interface elements in single detailed background image imported from Photoshop. Transparent SuperCard button were then overlaid to provide interaction.

The application was developed further to support multiple displays and inter-element messaging, e.g. pressing a 'power shut-off' buttons would inhibit the function of the rest of the panel components. This allowed the panel to be used by the group to explore the control aspects of panel design, such as fail safe systems, automatic default reset, etc. The Interface application was later adapted as a teaching tool for prototyping and panel design, (appendix H).
5.4 The Database Application in Vehicle Ergonomics

5.4.1 System Structure

The test applications provided a precedent for the development of the DAVE system by demonstrating the scope of the selected technical solution. Having shown the possibility of development based on the suggested lifecycle it was necessary to plan the proposed product. The first task was to identify an appropriate structure for the system. Its name, the Database Application in Vehicle Ergonomics (DAVE) had been arrived at around the time that the option of in-house development was considered as a viable option for group support. The structure of the DAVE system was dictated by the breadth of the group's requirements, as represented by the product specification, and by the project's limited resources. By taking a modular approach to development, the aim was to deliver functional products in stages rather than deliver one all-encompassing system at the end of the project. The modular approach had an organisational precedent in the form of an existing software tool within the establishment, the Modular Applied Physiology System (MAPS) (Wadsworth 1991). The structure of MAPS provided a convenient example on which the developer could base a concept of the final product.

The MAPS had been developed to support the work of the establishment's applied physiology group. Users were able to connect to the MAPS by text based terminals, linked to the establishment's VAX computer. The system provided a text based interface to the modules of the system, arranged in a multilevel hierarchy of system screens. The MAPS modules were indexed and accessed through a central entry screen.

Although adopting a similar structure, the DAVE project went on to explore alternatives unavailable to the developer of the MAPS. Assessment of the MAPS revealed usability problems stemming from the restrictions enforced by its low resolution, text interface. By selecting a desktop computer platform and a graphically orientated development tool, the project was able to pursue solutions to the MAPS interface problems. Whilst rejecting a text based interface, the modular structure offered a manageable approach for the individual developer. It allowed progress to be made in stages, delivering completed products sequentially throughout the project.
The DAVE system adopted the MAPS approach of providing user access to the various modules through a central entry screen, (figure 5.6). This entry screen could then provide the user with an index of available modules.

Figure 5.6 DAVE System Structure

5.4.2 DAVE Entry Screen

An entry screen was prototyped to provide direct access to a modular DAVE system. Its interface simulated the buttons and LCD screens of a hardware panel, (figure 5.7). This provided a metaphor for operation based on a familiar object within the user populations working environment. Use of a metaphor as an aid to usability spread from the desktop metaphor of the Star system (Smith et. al. 1990) to dominate the personal computer market by the end of this project.

The entry screen's design was based on the user approved graphics and code developed for the Interface test application, (figure 5.5). The user group's positive response to the 3-Dimensional effect interface of the Interface test application led to its adoption as the preferred interface style for the DAVE system. All members of the group recorded their preference for this interface approach over the 2D approach which was the commercial norm at the time of test application development. This raised button effect has since been widely applied throughout the commercial software market.

The interface advertised the modular structure of the system by listing the titles of proposed modules on functional buttons. As the entry screen was constructed before development of the first module, the animated buttons simply responded by presenting module proposals in the adjacent window.
The 3D buttons provided the user with a visual cue to the function of each interface area. This link between appearance and function was observed to be particularly important for those members of the group without previous computer experience. It was observed that these individuals frequently failed to make the connection between appearance and function for many 'standard' interface components such as scroll bars and menus. However, they were quick to link a button's appearance to the fact that pressing it would initiate a function. The animated effect and auditory feedback associated with pressing a button provided reassurance that a function had been selected. Users showed a tendency to press interface components repeatedly when no feedback was received.

Figure 5.7 DAVE Entry Screen
5.5 Discussion of Methods

5.5.1 The Contribution of Test Application Development

Whilst the cost associated with test application development may be estimated, (section 10.5.4), the value of this process could not be quantified in the same way. The central purpose of test application development was the confirmation of the development tool's scope. Their development would have been justified if they had shown SuperCard to be unsuitable for the proposed project. This would have allowed selection of an alternative approach early in the project. However, the process endorsed SuperCard as a suitable prototyping and development tool. Whilst this does not diminish the validity of the initial aim of test application development, it does raise the question of whether the time intensive process was justified as simply a safeguard.

However, the value of the test applications to the project goes beyond confirmation of functionality. In addition, their development was considered to provide:

1. Development tool experience.
2. An indication of project scope.
3. A code and graphics standard.
4. Project/work programme support.
5. A prototyping process model.

5.5.2 Development Experience

A key argument for the selection of SuperCard as the project development tool was its apparent ease of use. Despite this there was the necessity for the developer to gain experience in applying the tool. Tool confirmation through test application development provided this opportunity. Nielsen et. al (1991) reported the ability of a student group to handle the basics of HyperCard programming within two days in comparison with the two months required for another more formal programming language.

Whilst it was possible to construct basic applications immediately using the SuperCard tool, a growing sophistication in its implementation was apparent in the interface and
code quality of subsequent test applications. The testing of the tool's functionality provided a suitable approach to providing this experience. This development period enabled the developer to tackle a wide range of programming problems associated with handling text, numbers and graphics. Development experience from the first two test applications led to the more efficient approach of breaking code processes into smaller routines. These routines could then be copied to other programs or interface elements which required the same function, e.g. control of feedback following button interaction. This improvement in programming technique to suit the tool grew from experience in applying SuperCard.

If direct development had been undertaken the errors made during test application development would have emerged in the design of the first of the project's specified products. This carries a risk associated with the proposed prototyping approach to development. As the intention was to evolve prototypes towards a final product, poor design decisions related to a lack of experience could become embedded in a final product. This risk was reported by Bickford (1994) who advocated rapid prototyping in which the prototype was discarded. Whilst problems were still encountered during the development of the DAVE system software, the preceding development experience provided an improved position to anticipate problems and tackle them when they arose. It is concluded that if the project had demanded the immediate development of a user accountable tool, the product's quality would have been lower than the eventual Anthropometry module. Lower interface standards or reduced code robustness may have prejudiced the users against the project, despite promises of improvement.

5.5.3 Indication of Scope

The concept of in-house software development presented many uncertainties for the management of the project. Over ambitious product specification based on a partial understanding to the tools and resources of the project could have lead to project failure. Achieving a common concept of a proposed system is important when specifying the details of its design. An effective method of achieving this common concept is to provide the users with a model of the proposed product in the form of a prototype (Spence 1991, Hekmatpour and Ince 1987, Luqi 1989). These observations were repeated in the VDS project. User experience of the test applications was seen to lead to increasingly informed and productive discussions. The development of these applications, leading up
to the formal group meeting, helped to align the users' expectations and the consequent specification with the realistic scope of the project.

The test applications demonstrated the potential for developing robust, fully functional applications. They revealed the effort associated with development tasks, in addition to simple coding. As noted in sections 5.3.2 and 5.3.3, the effort associated with data gathering outweighed the programming time load. Recognition of the project's limited resources of a single developer, and the precedent of the MAPS system, (section 5.4.1), led to the project's modular approach to the development of an information support tool for the group.

5.5.4 Product Utility

Whilst the test applications were intended as simple demonstrations of SuperCard functionality, development effort extended beyond this immediate goal. The Text application was applied to project specific abstracts cataloguing. The Numerical application provided the data formatting necessary for the development of the DAVE system's first module. As a result, both of these applications contributed to the development project.

The development of the Interface application continued after formal selection of SuperCard. Further development effort was justified by the proposed product's enhancement of one area of the group's capability in terms of their regular work programme. Whilst this diverted development effort away from the main project goals, several code and graphic elements developed for the interface application were eventually became standard components in the project's main software.

5.5.5 The DAVE Entry Screen

The entry screen acted as an interactive design proposal for the DAVE system. It indicated the proposed modular structure and provided the users with an impression of the suggested interface approach. It presented these features to the group in an accessible form. The user group demonstrated an understanding of the screen's basic functionality and expressed satisfaction with its form and function.
The project's adoption of a modular structure was helped by offering a means of dividing the project into achievable sections which could contribute to a gradually broadening tool for the group. At the same time the structure of the DAVE system was evolved. Both the technology supporting the tool and the way the tool was applied changed over time to make the entry screen redundant as a central link to the functions of the DAVE system, (section 9.3).

Whilst the entry screen was a separate application, the formal lifecycle proposed for module development was not applied to the development of the entry screen in its own right. However, it did follow the prototype development cycle in which its elements were arranged before functional code was developed and the graphics refined. In addition, user feedback was used to iterate the design. The addition of an 'LCD' information screen was a response to user reaction to the first entry screen proposal. Although detailed specification was not undertaken, it was evaluated along side the Anthropometry module before the initial delivery of the DAVE system, (section 6.5.2).

5.5.6 The Prototype Development Cycle

The development experience gained over this period provided a guide to future development using this tool. By encapsulating code within SuperCard's hierarchy of components the developer was able to maintain a clear picture of the structure and function of the application and its components. The application's 'start-up' instructions were placed in the 'project area'. Main control functions, such as database searching and result display, were placed in the 'window area'. Commonly accessed functions, such as monitoring the cursor position, were placed in the 'background area'. Finally, interface code was placed in the associated interface objects. This encapsulation of small functional routines suited the initial design cycle, (figure 5.2) in which functions were written, tested and iterated separately. This approach was used successfully to develop the functional elements of all three test applications.
What distinguished the development of the Interface application from the other test applications was the initial creation and arrangement of non-functional interface elements within the application window. This allowed the application to be assessed as a whole, in the same way a paper prototype might be assessed, before work began on adding functionality to the system. The technique was then extended by the ability to continue the iterative design of each element to include functional code. In comparison, the Text application design approach involved the sequential addition and full design of elements to whatever interface area remained. The result was a design which incorporated the full specified functionality without taking account of its overall appearance and usability. The rapid pre-design of the interface test application helped to avoid this sort of design problem. Its two stage approach, applied to the final test project, was considered to be more appropriate for the development of the DAVE system. This approach is illustrated in a model designed to guide prototype development for the DAVE system, (figure 5.8).
The prototyping design model included three distinct iterative cycles, A, B and C. Cycle A involved interface design and evaluation at a high-level equivalent to paper based rapid prototyping. This provided the starting point of the development cycle with the specification of application components, including functions and interface elements. The interface elements could then be arranged within the application window. This provides the first concrete representation of the proposed product which could then be assessed in terms of content and usability. This level of assessment might lead to re-specification in which a new element would be added or an existing one modified or removed. The consequent interface modification would prompt re-evaluation of the design and a possible repeat of the iterative cycle. Once an acceptable non-functional design was produced the process model would move on to cycles B and C.

Cycle C was the original iterative loop for the design and testing of application elements, as applied to the development of the first two test applications. An element created in cycle A, such as a function button, would be selected for development. Its associated graphics and controlling code could then be created, tested and iterated.

Cycle B followed the acceptance of each component's design. The cycle returned to an assessment of the overall prototype to consider the effect of the new element's design and function. Acceptance of the overall design would allow the selection of another component for development. If assessment of the overall design revealed possible improvements, the process would return to cycle A in which the overall design including its components and layout could be changed. Completion of the 'prototype development cycle' was at the point where all elements had been developed and cycle A didn't demand any further elements or design changes.

The form of the product of the prototyping stage distinguished this model from the rapid prototyping model (figure 3.4). In the rapid prototyping model the design cycle was followed by the generation of a detailed specification document. This document would support the transfer of the prototype's design to a product developed using a separate programming tool. The proposed model skipped the transfer between prototyping and development languages by using the high-level development tool for both iterative prototype development and construction of the deliverable product. This avoided time cost associated with the document generation and the application coding exercise. The
viability of this approach was dependent on the chosen tool's ability to fulfil the role of both a prototyping tool and an implementation tool.

Whilst SuperCard's ability to evolve and deliver software applications was demonstrated, the design of each application was developer centred with the addition of informal user input. As one of the highlighted advantages of in-house development was user/developer contact, it was important that the DAVE project built on this close working environment. In addition to providing a guide to development, the prototype development cycle provided a model with which appropriate points for user involvement in the development process could be planned. As the product was to be developed within a working environment it was important that the development process presented a minimal conflict with the users' regular work programme. The form of user input to the test application development process suggested that the most appropriate point of user input within the cycle would be at the point of overall evaluation, at the end of cycle A.

Formal user involvement in the parallel analysis of requirements presented the group members with a conflict between contributing to the development project and fulfilling their regular work commitments. Mutual times for formal meetings proved to be difficult to arrange. In contrast, informal input to test application development was able to fit around existing commitments. As a result, it was proposed that informal user involvement should be used for system level assessment within this design cycle. The major drawback of this approach was seen to be its reliance on the willingness of group members to give their spare time to the development project. It was also possible that some group members would provide a greater input to the design of the system than others.
5.6 Conclusions

The development of test applications started with the initial assessment of the software development tool which expanded into small development projects in their own right. In doing so, they exceeded the study's objectives, whilst continuing to meet the project's overall objective of supporting the user group. Conclusions drawn at this point in the project were as follows:

1. The test applications confirmed the ability of the development tool to deliver a robust application through design evolution.
2. Developer experience in applying a new tool supported the main project.
3. The 'hardware panel' application provided a user approved interface style for the DAVE system.
4. The development effort applied to code and graphic design was spread across the full project with their refinement into robust, transportable elements which were incorporated in later applications.
5. Test application development provided an introduction for novice members of the group to computers before the need for them to specify their requirements.
6. The 'prototype development cycle' matched the development tool to guide design exploration within the development process, as required by the approach of minimal specification.
7. The development effort for each test application was offset by their direct support for the main project or the group's regular work programme.
8. Test application development supported project specification by providing product models as a guide to the determination of attainable goals.
6 The Anthropometry Module

6.1 Chapter Overview

Chapter six presents the project's first complete development cycle. It describes the design, evaluation and implementation of the first specified module of the DAVE system. The purpose of this module was to provide the user group with access to the establishment's anthropometry database.

The chapter sets out the proposed lifecycle for the development of the Anthropometry module. The choice and application of each lifecycle process is described and assessed. The minimal product specification was advanced by a further module level specification. Data set selection and requirements definition led to the construction of a function list, which provided a starting point for prototype design. Design alternatives for the module were explored through the prototype development cycle which was set out following the development of the test applications.

A discussion of methods considers the approaches of paper based survey, semi-structured interview and informal discussion for the functional specification of the Anthropometry module. The contribution of user feedback in the iterative development of the module is presented in relation to the influence of expert evaluation and formal user testing. Also considered are the issues related to the technique of background logging for determining usability issues.

These findings are incorporated in the redesign of the lifecycle model. The revised model extends the development sequence to include module specification and suggests a separate maintenance phase in addition to the development cycle. The chapter concludes with a list of recommendations which are tested in the development sequence of the subsequent module of the DAVE system.
6.2 Module Development Lifecycle

The preceding chapters discussed the project's background, leading to the concept of a modular product. At the start of the project, the selected lifecycle included a separate 'coding' activity. The acceptance of a 'prototyping tool' for both product design and delivery allowed 'coding' to be removed from the lifecycle model, (figure 4.7).

The first phase of the proposed lifecycle model produced a minimum critical specification, (Cherns 1976), and provisionally selected technical tools to explore and deliver the concept product. Development of the test applications confirmed the tool's suitability and provided a precedent for development using the tool. The process of tool confirmation allowed the determination of a prototyping process model as a guide to product development, (figure 5.8). The prototype development cycle was taken as the central component of a lifecycle to guide the development of the DAVE system. This cycle was considered to be the key to meeting the project's user centred design philosophy. The emergence of these factors led to a re-evaluation of the lifecycle before development of the project's first deliverable product. The revised model is presented in figure 6.1.

Minimal Specification

Prototype Evolution

Acceptance

Delivery

Maintenance

Figure 6.1 Evolutionary Development Lifecycle

The revised model confirms the bypass of a coding phase due to the demonstrated capability of the development tool. SuperCard proved to be flexible in all stages of design evolution leading to the delivery of robust standalone applications, (section 5.3). The second lifecycle change was the addition of a formal acceptance phase. The lifecycle presented in figure 4.7 followed the prototype/test cycle with implementation and

110
maintenance phases. The revised model included formal acceptance as a defined phase to ensure complete acceptance across the group before final delivery. This quality check was included to account for looseness of the specification and the lower level of control associated with informal user assessment during test application design.

The lifecycle model was designed to provide a development sequence. It did not define the detailed processes required at each phase of the cycle. Whilst the lifecycle was able to draw from the lessons of test application development, other issues such as user involvement and formal accountability needed to be addressed in the development of deliverable modules. Beyond the proposed prototype development cycle, the choice of lifecycle processes depended on the aims of each phase and the available resources.

The model's first phase produced a minimum critical specification in the form of a priority list of development areas. The absence of detail reflected the group's inability to fully define possible products. Rather than attempting to force the users to meet the demands of detailed specification, the lifecycle accepted the limitations of the scenario. Instead, it adopted a user centred strategy for detailed specification based on prototype evolution.

The test applications provided the model for development of the first system module. They demonstrated the combined ability of the prototype development cycle and the development tool to explore and match design solutions to loose product concepts. Recommendations following test application development suggested the complete design of a non-functional prototype followed by the iterative development of the design's components. Design of the Text and Numerical test applications started with a function list. This provided an intermediate between their broad requirements definition and the initial prototype design. This approach was retained for the design of the Anthropometry module as an aid to the initial non-functional prototype design.

The benefit of continued use of informal user involvement in the design process was suggested, (section 5.5.6). The risks of disproportionate representation and user disinterest were countered in the revised model by the inclusion of an acceptance phase before delivery. The component processes of formal acceptance were decided during the period of Anthropometry prototype development. Their selection was based on the form of the evolving product and the availability of subjects to evaluate the system.
Similarly, the processes of delivery and support were matched to the project's characteristics as they became apparent.

6.3 Anthropometry Module Specification

The lifecycle's specification phase highlighted anthropometric data as an important reference for the group, without detailing how this information should be presented. In this high level form, the 'system level' specification did not provide sufficient detail for the prototype development phase to design a solution. There was a need to start the prototype development phase with a further specification process in which user requirements for the product could be defined in more detail.

6.3.1 Data Set Selection

The 'system level' specification process identified the establishment's anthropometric database as the user group's priority for computer based support. The data had been located and partially acquired during the development of the Numerical application, (section 5.3.3). Consideration of the data confirmed that it contained a range of measures and populations which related to the group's work programme. These factors led to its selection as the module's proposed data set. If the proposed data source had been inaccessible other sources would have been sought.

Review of the available data revealed 80 measured dimensions and 8 subject populations. As further anthropometric measurement fell beyond the scope of the project, the form and range of this data set limits on the functional scope of the proposed module. User discussion, as part of the subsequent requirements definition activity, confirmed that the group needed a subset of the available data. As the project's goal was to tailor the product to the needs of a defined user group, it was possible to reduce this list to data which was specific to the group's work programme. The aim of condensing the list was to provide the design option of simplifying the prototype's user interface by removing redundant information.

Each member of the group was provided with a comprehensive listing of available measures and surveys and asked to indicate which anthropometric measures they
considered necessary for their work. The subjects crossed off any measures or survey groups they considered unnecessary for the specific work of the group. All measures selected by one or more users were included in the final list.

6.3.2 Requirements Definition

It was important to determine how the specified anthropometry data was used and in what form it was required by the group in order to design the functional product. This activity is referred to as the 'requirements definition'. The approach of semi-structured interview of the user group was chosen to define the functional requirements of the module. The group was interviewed individually to ensure consistent levels of input. Each interview was loosely structured to avoid constraining the discussion, as risked by a fully structured interview. The questions raised issues concerning the circumstances in which anthropometric data were used and the form of each enquiry. The product of these discussions was the breakdown of the group's use of anthropometry data into two model enquiries.

A. What is the value of a specified body dimension for the Nth percentile of a given population?

B. What percentage of a given population have a body dimension above or below a specified value?

These model questions defined the Anthropometry module's functional aims. Consideration could then be given to what functional elements an application would require to allow the user to answer these model questions.

6.3.3 Function List Construction

A list was used to define the functions each test application needed to confirm SuperCard as the chosen development tool. In doing so, these function lists defined each prototype's core components. For the Anthropometry module, the list recommended functions necessary to answer the questions set out in the requirements definition. This provided the first design step in the development of the prototype. The functions were set out in a
list before design to ensure that all important elements were included in the initial non-functional prototype design. This allowed issues such as functional grouping and the balance between elements to be addressed before focusing on detailed design issues such as functional code. The function list is presented in table 6.1.

<table>
<thead>
<tr>
<th>Application Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>11</td>
</tr>
<tr>
<td>12</td>
</tr>
</tbody>
</table>

Table 6.1 Anthropometry Function List

Functional elements 1 and 3 of the list represent the display of the answers to the two model questions. Answering question A required the fixing of elements 4, 6 and 8. Answering question B required the fixing of elements 2, 6 and 8. The remaining listed elements included a combination of required features and proposed enhancements to the basic functionality of the application.

6.4 The Prototype Development Cycle

6.4.1 The Non-Functional Prototype

Design of the Anthropometry module followed the prototype development cycle, (figure 5.8), as applied to the development of the test applications. The key difference was the addition of user involvement.
As recommended, (section 5.5.6), the early prototype took the form of a non-functional interface design. The design included the main interface components required to meet the functional definition. The design used the reduced set of measures to list all vehicle specific measures on screen. The title of each measure was associated with a field to display its value for a selected percentile. All measures were presented on a single screen to avoid the navigation problems associated with multiple screens, as observed with the MAPS software. A potential problem with this approach was the increased complexity and reduced legibility of screen elements.

Informal group discussion highlighted additional functional requirements and possible enhancements to the prototype's design based on this initial design. These included:

- Comparison of measures for different percentiles (Specifically 5th and 95th).
- Finding values when the measure name is not known.
- Finding details of the measurement technique used for specific dimensions.
- Determining the number of data points.

Continuing cycle A of the prototype development model, (figure 5.8), new elements were conceived to meet these additional requirements. These elements were inserted in the non-functional prototype leading to further changes in the overall design.

Review of this prototype by the developer and the users did not raise further requirements at this point. This allowed the design process to move on to cycles B and C of the prototype development model select and develop each element in detail.

Figure 6.2 shows an early prototype screen including the specified module elements.
6.4.2 Design of Functional Elements

The prototype was used as a tool to help the developer and the users to consider all aspects to the prototype's emerging design. This led to design iteration following the prototype development cycle, (figure 5.8), with input from both sides.

An example of iterative design is provided by the introduction and evolution of graphical searching. Despite appropriate grouping of the reduced list of measures, the visual search for measures was still found to be difficult by the group members. Following cycle A, recognition of this problem was followed by the suggestion of a design solution. The functionality of the prototype's illustrative photograph, (figure 6.2), was expanded to allow measure selection by overlaying 'hot-spots'. These overlaid buttons allowed the user to select a measure by pointing to its position on the photograph.

Iterative cycle B was followed to develop the 'hot-spot' interface element. This design was explored in a developer centred iterative loop in which the design was repeatedly modified and tested, (cycle C). After completion of this functional element to the

Figure 6.2 Early Anthropometry Prototype
developer's satisfaction, the revised design was shown to the users to gain feedback, continuing cycle B. The impact of this addition to the overall design was then considered by the developer and the users by interacting with the semi-functional prototype.

Alternatives at this point in the model included cycle A through the specification of an overall design change or cycle B through either the selection of another element to develop or the redesign of the 'hot-spot' element. Whilst the approach of graphical selection was considered an improvement, user interaction at this stage revealed confusion concerning which 'hot-spot' related to which measure. Consequently cycle B was followed to redesign the 'hot-spot' element. The new designs explored through cycle C distinguished between neighbouring 'hot-spots' by indicating the direction of the associated measurement, (figure 6.3 and figure 6.4). The return to design review by the users, (cycle C), informally accepted this solution.

![Figure 6.3 Photograph 'Hot-spots'
![Figure 6.4 Directional 'Hot-spots'

6.4.3 Interface Form and Structure

Having established a suitable form and layout for the module, the hardware panel style was applied, (figure 6.5). The module used a single background graphic incorporating each of the prototype's interface elements, as recommended following test application development, (section 5.3.4). Buttons and fields were made transparent and overlaid on the background providing immediate functionality to the new design.

The code contained within the overlaid elements was written to function independently from its position within the interface. This approach allowed flexible redesign of the interface style and the arrangement of its components. Once the interface had been transferred to a single background graphic, interface redesign involved modification of
the graphic within the drawing package followed by the repositioning and/or scaling of the transparent overlaid elements.

Figure 6.5 Anthropometry Prototype

6.4.4 Completion of the Prototype Development Cycle

Until this point, prototype iteration had been developer driven, with the support of unstructured user involvement. The informal nature of user involvement produced several effects. User input was frequent and untimetabled, fitted around work commitments. However, the input was disproportionate with interested individuals providing more input to the design than the others. Those users who became actively involved in the development process developed a better understanding of the function and scope of the overall system through their experience of the current module's evolution. This was evident in the larger contributions of a core of informal users during group discussions.
A major problem encountered in the prototype development cycle was the selection of a point at which to transfer between the development phase and the formal acceptance phase. An extensive list of minor improvements could have been proposed and applied to the design of the module, dragging out completion time. The decision to initiate the acceptance phase rested with the developer. The criteria chosen to guide this decision were as follows. The prototype must:

1. Provide the specified functionality.
2. Be easily accessible to the specified population.
3. Provide accurate and unambiguous information.
4. Be consistent and robust.

Transfer to the lifecycle's acceptance phase depended on the confidence of the developer in the ability of the product to meet these criteria. The level of confidence relied on the developer's detailed knowledge of the product and the informal responses of the users.

6.5 Acceptance

User feedback within the prototype development cycle provided the group with control over the product's design. The purpose of an additional acceptance phase was to overcome the risks of informal control such as uneven group involvement and the subjective nature of the feedback.

Two approaches to prototype evaluation were selected for the acceptance phase of the Anthropometry module. These were:

1. Expert evaluation - To draw on the experience of another developer to highlight any design issues overlooked in the prototyping cycle.
2. Formal user testing - To provide an objective measure of the whole group's ability to use the product.

These approaches aimed to both highlight possible improvements and to ensure acceptability before delivery of the product. The acceptance phase was designed to follow each formal evaluation with the opportunity for product iteration, (figure 6.6). Expert evaluation was to be completed first, implementing any design recommendations before
formal user testing. Any further design issues raised by formal user evaluation would then be addressed before delivery of a final product.

Figure 6.6 Acceptance Phase

By anticipating design problems and including an opportunity to iterate the design, the implications of problem discovery were reduced. Acceptance became an extension of the development phase in which objective measures were used to direct the further improvement of the product. This approach relied on the flexibility of the development tool to handle design changes late in the development lifecycle.

6.5.1 Expert Evaluation

Following completion of the Anthropometry module's specified functionality, the full system was evaluated by an expert in the field of software development. The individual chosen was the developer of the MAPS software, (section 5.4.1). This provided the expert with a good understanding of the establishment, whilst not being closely
associated with the user population or the DAVE development project. The evaluated system included both the DAVE entry screen, (figure 5.7), and the Anthropometry module's single screen, (figure 6.5).

The expert was given an overview of the work of the target population and the specific aims of the module as represented by the functional definition. The system was demonstrated by the developer and the expert allowed to conduct a self paced evaluation. The expert was asked to comment on issues relating to both design and functionality throughout the evaluation. Issues raised were recorded by the developer. Following the evaluation, these issues were discussed between the developer and the expert. This allowed the clarification of the expert's recommendations and the resolution of disagreements. The expert's recommendations are listed with the design actions in table 6.2.

<table>
<thead>
<tr>
<th>Recommendation</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow alteration of units of measurement.</td>
<td>A button was added to toggle between metric and imperial units.</td>
</tr>
<tr>
<td>Display units of measurement.</td>
<td>Units were displayed on the button indicating current unit.</td>
</tr>
<tr>
<td>Allow addition of clothing increments.</td>
<td>The data required was not available. Would be added in the event of the data becoming available</td>
</tr>
<tr>
<td>Provide additional anthropometric measures.</td>
<td>The data required was not available. Would be added in the event of the data becoming available</td>
</tr>
<tr>
<td>Reference the source of the data.</td>
<td>Survey dates and relevant information was presented in the message window when selecting a survey group</td>
</tr>
<tr>
<td>Include survey summary statistics.</td>
<td>Considered unnecessary by the group. Added in the module's mid-life redesign.</td>
</tr>
</tbody>
</table>

Table 6.2 Recommendations and Actions following Expert Evaluation
6.5.2 Formal User Evaluation

6.5.2.1 Aims

The process of formal user evaluation of the Anthropometry module aimed to:-

1. Determine each group member's ability to use each of the module's features.
2. Highlight non-intuitive aspects of the design.

The Anthropometry module represented the group's only source of access to the establishment's database. The baseline for acceptability was not speed of access, in comparison with existing methods, but the ability of each specified user to retrieve accurate data. It was recognised that access to the system would be occasional. Consequently, the priority of the design was to provide efficient access to the required data.

6.5.2.2 Method

The DAVE system, including the Anthropometry module, was installed on a standalone Macintosh computer in an office central to the members of the group. The user test was based on the completion of a self paced question sheet, (appendix B). The question sheet was distributed amongst the members of the group. The sheet included questions testing all aspects of system functionality. Each user was asked to complete the sheet in their own time, at their earliest convenience. The sheets were returned to the developer following completion.

Each question included a space in which the subject was asked to report any comments relevant to use of the system. These comments were correlated and any ambiguity was resolved by discussion with the subject. The question sheets were marked for accuracy. The module included code which logged user interaction in the background. The event log for completed question sheet was retrieved to determine the time spent on each question and the user's sequence of actions.

Completion of the question sheets without the presence of the developer was used in order to evaluate the module in its actual environment. Accuracy and background logging
provided quantitative data on which to base conclusions as to the suitability of the module for full implementation.

As use of the tool was expected to be occasional, its design concentrated on simplicity and accuracy. In order to test the module's accessibility for the occasional user instruction in the use of the module was not given prior to the test. Guidance was provided in the form of an on-line help system which could be accessed when required.

By not providing training or assistance during the test, the evaluation used a 'worst case' scenario for evaluation. It tried to reveal design problems as opposed to trying to 'pass' the product as acceptable. It avoided the assumption that an individual trained to use the system will continue to use the system efficiently in the future when access is occasional. The aim was to reveal non-intuitive features of the system which might not be apparent when the individual uses the tool regularly but which might occur if there are long gaps between uses of the tool. Although the users were not pre-trained for the test each had some experience of the prototype due to the approach of informal user involvement in the prototype development cycle.

6.5.2.3 Results

The question sheets were completed by each current member of the group over the period of three weeks. The sheets were marked and the comments assessed and clarified by discussion with the group members. The background log files for each session were gathered to determine user actions and completion time for each question. The results are shown in tables 6.3 to 6.5.

Table 6.3 shows the total completion time and errors for each subject. The table includes a control measure for completion time. This measure represents the time taken by the developer to complete the question sheet. It should be noted that the recorded completion time for the other group members does not allow for time taken to add comments to the sheet.

There was a positive response to the graphical interface approach with most subjects preferring to select measures using the photograph. The parallel highlighting of list measures and photographic illustration was considered to help the search process. Each
member of the group expressed confidence in searching for measures using the interface.

<table>
<thead>
<tr>
<th>Subject</th>
<th>Completion Time (sec.)</th>
<th>Question(s) Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control</td>
<td>481</td>
<td>-</td>
</tr>
<tr>
<td>A</td>
<td>1231</td>
<td>No. 4</td>
</tr>
<tr>
<td>B</td>
<td>1523</td>
<td>No. 10</td>
</tr>
<tr>
<td>C</td>
<td>1851</td>
<td>No. 10</td>
</tr>
<tr>
<td>D</td>
<td>1914</td>
<td>No. 7</td>
</tr>
<tr>
<td>E</td>
<td>2325</td>
<td>-</td>
</tr>
<tr>
<td>F</td>
<td>2451</td>
<td>Nos. 5, 10</td>
</tr>
</tbody>
</table>

Table 6.3 Question sheet results

The key measure of the ability of the group to use the module was the number and significance of the errors. The incorrect answers, recorded in table 6.3, fell into two categories of significance. The lower level errors were committed for questions 7 and 10 in which some users failed to provide any answer. On two other occasions the errors involved incorrect answers. One was non-critical error as the subject indicated that they realised the value quoted was for a different percentile to the question. The other represents a more important error as the subject was apparently unaware that their answer was incorrect. It was incorrectly answered due to a failure to reset the percentile column.

The comments sections of the question sheet allowed the users to provide formal feedback to the developer on all aspects of the system, (table 6.4). They found the simple interface of the DAVE entry screen easy to use. This was reflected in the access time which ranged between 3 seconds and 10 seconds, (table 6.5). It should be noted that these results do not include the completion times of the control.
<table>
<thead>
<tr>
<th>Question</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Easy to use.</td>
</tr>
<tr>
<td>2</td>
<td>Preferred visual searching.</td>
</tr>
<tr>
<td>3</td>
<td>No comments.</td>
</tr>
<tr>
<td>4</td>
<td>Selected wrong hot-spot but realised due to list highlighting.</td>
</tr>
<tr>
<td>5</td>
<td>Help button used to guide percentile change.</td>
</tr>
<tr>
<td>6</td>
<td>Associated measure selection with display in the 'measure box'.</td>
</tr>
<tr>
<td>7</td>
<td>Didn't know how to remove a measure from the sheet to be printed. Didn't know if printing had been successful.</td>
</tr>
<tr>
<td>8</td>
<td>Help button used to guide change of survey data.</td>
</tr>
<tr>
<td>9</td>
<td>Survey recalculation can't be cancelled.</td>
</tr>
<tr>
<td>10</td>
<td>Couldn't see how to answer the question.</td>
</tr>
</tbody>
</table>
| General  | Too close to the screen.  
Would like clothed data.  
Liked demonstrated help sequences.  
Guided printing sequence was annoying.  
The help button is not obvious.  
Stool height measure required.  
Buttock to heel length representation of photograph was confusing. |

Table 6.4 User Comments

Usability problems were mainly encountered by the three older members of the group. These individuals each had little experience using computers. Their completion times tended to be longer due to a more deliberate approach to interaction and the regular referral to the on-line help system. Despite their tentative approach, these users did not make more errors than the rest of the group. Conversely, one of the computer novices was the only subject to answer all questions correctly. The other members of the group showed a greater tendency to learn by interacting with the functions. Recognising that the test was completed without prior experience of the system, it was anticipated that users would become more efficient with experience of the system.

Analysis of the action sequences showed that, with the exception of printing, most subjects were able to complete the search tasks by following an optimal interaction sequence. Examples of longer operation sequences were generally due to multiple referrals to the help system.
Table 6.5 Question Completion Time and Action Sequence Ranges

6.5.3 Design Iteration

The evaluation highlighted several cases of inappropriate design. The design problems and the project's actions are listed in table 6.6.

None of the problems revealed prevented the specified user group from using the module to accurately retrieve the anthropometric measures in the form specified by the original functional definition. The listed actions were taken to correct the problems encountered. The iterated module is illustrated in figure 6.7, in the form taken by the module at the point of delivery.
<table>
<thead>
<tr>
<th>Problem</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Printing</td>
<td>Method was altered to print the entire screen rather than selected measures.</td>
</tr>
<tr>
<td>Finding 'More' information</td>
<td>Information was automatically presented about measurements and populations when the cursor was positioned over the relevant button.</td>
</tr>
<tr>
<td>Poor Help icon design</td>
<td>The &quot;?&quot; Icon was replaced by the text &quot;HELP&quot;.</td>
</tr>
<tr>
<td>Incorrect 'Buttock-to-Heal' illustration</td>
<td>A correct photographic representation was provided in the mid-life interface redesign, (figure 9.4).</td>
</tr>
<tr>
<td>Absence of 'stool height' measure</td>
<td>Stool height was added to the interface.</td>
</tr>
<tr>
<td>Poor workplace design</td>
<td>Acceptable layout designed for the implemented system.</td>
</tr>
</tbody>
</table>

Table 6.6 Design Problems and Actions following Formal User Evaluation

![Implemented Anthropometry Module](image)

Figure 6.7 Implemented Anthropometry Module
6.6 Delivery

The acceptance phase was followed by the lifecycle's delivery phase. Its goal was to install the product within the user working environment. The DAVE system was intended to be a standalone tool. This reduced the task of delivery to the installation of the tool on a single computer, located in a central office. The system’s move to a network based tool, (section 9.3.3), expanded the task of the delivery phase to include wider distribution.

The lifecycle model, (figure 6.1), retained the Waterfall model's maintenance phase as a final activity. This phase was distinct from the other lifecycle phases as it did not have a defined end point. The anticipated maintenance requirement for the delivered product was divided into two categories. The first was the correction of minor design faults following delivery. The second was a need to provide long term support for the product, such as adding to the module's database following a proposed survey update. In response, the proposed process model, (figure 6.8), included an iteration cycle to correct problems discovered by the users immediately after delivery. This was proposed as a limited period after which the cycle would return to system specification and consideration of the next stage of the project's work programme. The remaining maintenance task of long term support was excluded from this cycle as a separate task. The position of the ongoing software support task was addressed in the design of the overall model, (figure 6.9).

Although allowed for within the delivery model, the iterative process was not called upon for the Anthropometry module's development project. The anticipated wave of maintenance requests did not occur. The delivered module proved to be robust over the course of the project. The simple distribution and absence of initial maintenance tasks reduced delivery from a 'phase' to an 'event' in the lifecycle.
6.7 Discussion of Methods

The study was longitudinal, leading to a continual review of the process as a guide to continued development. Whilst there are several references which suggest that a prototyping approach may be effective for specification and design, (Gomma 1983, Harker 1988, Luqi 1989, Smith 1991), few discuss the finer details of the design process. The study responded by designing processes and selecting techniques considered to best fit the project's stated intention of user centred design through prototype evolution.

The starting point for an evaluation of the lifecycle and its constituent processes must be an assessment of the product. At this point in the project, judgement of the Anthropometry module relied on the results of the formal evaluations and the level of maintenance required immediately after delivery.

The Anthropometry module achieved its defined aims, as confirmed by evaluation based on a 'worst case' scenario. The user centred design process supported the evolution of a graphical interface which allowed the defined range of users to accurately retrieve the
required data. In the short term, the product was found to be acceptable to the user group. A long term assessment of the module is presented in chapter 10 based on the usage and development cost data gathered over the period of the project. This provides an objective measure of the relative success of the module.

6.7.1 Module Specification

Following the traditional lifecycle model, adopted at the start of the project, the intention was to follow the specification phase directly by prototype development. However, the specification provided by the group discussion was limited to a top level description of the proposed product. Further detail was required at the start of the design process. A set of activities were undertaken to provide sufficient information to make a start on the prototype design. The activities of data set selection, requirements definition and function list construction combined to form a separate 'detailed specification' phase prior to the formal prototyping phase.

Data set selection was the starting point for specification with the identification of the establishment's anthropometric database for support. This activity was returned to following requirements definition to further tailor the module to the requirements of the user group. Requirements definition used the group members as the information source for determining the data which might be used. The function list, retained from the test application development approach, proposed a set of functions with which to build an initial prototype.

An important distinction between the design of the entry screen and the design of the Anthropometry module was their relative complexity. The entry screen's role was simple in comparison to the large variation in possible Anthropometry module functions and designs. The informal approach to the design of the entry screen, was not appropriate for this more complex application.

The techniques chosen for group survey to specify the Anthropometry module were questionnaire completion and semi-structured interview. Each matched the goal of the phase. Semi-structured interview allowed the users to freely discuss their requirements whilst retaining consistency across the group. The questionnaire allowed
simple selection of available measures and populations at the convenience of the individual. These formal techniques involved the users in the module specification phase, making the process more accountable to the users. They confirmed the continuing user role in the project's design process.

The group's lack of experience with computer based systems was apparent in their uncertainty when asked to discuss the future module's form and content. Their difficulty lay in the absence of examples on which to form an impression of what was possible in the development of the module. Their strength was found to be their knowledge of the application domain. By directing specification away from a description of how the module should function, to how the data might be used returned the focus to the users' area of expertise. This allowed the users to consider not only how they currently used anthropometric data but also to consider alternative ways of applying the data. This was illustrated by the two requirement definitions. Definition A described the normal way of using tables of anthropometric data. Definition B represented an enquiry which, although applicable to the work of the group, was difficult to answer using the existing data sources.

Construction of the function list was a deductive process carried out by the developer in relative isolation, Application elements were matched to the requirements definition without concern for interface structure and layout. The benefit of this approach was as a guard against omissions helping to balance the interface elements in design of the initial prototype. In addition, it provided a lifecycle deliverable which aided the management of the project. Whilst the function list helped to break down the interface enquiry task into its component parts, this task might have been achieved by directly prototyping the application. This questions the need for functional list construction as part of the design process. It was concluded that despite limited evidence of its benefit for this module, the need to balance interface elements would increase for more complicated applications.

Despite seeking user views concerning possible design solutions, usable feedback was not provided until the specification was translated into the visual form of the prototype. This highlighted the difficulty experienced by the group in forming a concept of the proposed product without an example on which this could be based.
6.7.2 Module Development

Anthropometry prototype evolution followed the prototype development cycle, (figure 5.8). Initial prototype construction was based on the translation of the function list into basic prototype elements which were arranged and modified to explore interface solutions. Interface exploration revealed several possible approaches to the design of the tool. Investigating the alternatives within the prototype window provided a visual illustration of possible approaches. Presentation of the various concepts in an accessible form to both the developer and the users promoted mutual understanding and effective communication. This supports Smith’s (1991) suggestion that prototyping offers an advantage over technical documentation for the purpose of specification.

User input to the prototype development cycle took place at a ‘top level’, (cycle A). The decision concerning which design alternative to advance further was retained by the developer, guided by both user views and knowledge of the development environment’s scope. The software model provided by the prototype helped to stimulate user interest in the design. The resulting informal discussion led to a refinement of the diverse design concepts represented by the prototype, and provided a check against subject matter errors.

Jasany (1990) observed that rapid prototyping helped the designer to concentrate on design solutions by delaying the need to solve programming problems. Design of the Anthropometry module provides a good example of how prototyping was a supportive medium for creative design. This study goes further by suggesting a structure, in the form of the prototype development cycle, in which both the developer and the users were able to contribute to an evolving design. The function list acted simply as the starting point to design. Within the prototype development cycle the only constraints to design were the inclusion of standard components and the boundaries of the development environment. Prototyping allowed elements to be simply added and removed. If assessment of the prototype showed an element to be inappropriate it could be redesigned or removed.

An example of the prototype development cycle’s support for broad input to a creative design process is provided by the evolution of the module’s illustrative graphic. Although the graphic was not specified, it evolved to become the group’s preferred
means of accessing the module's associated database. The conception and design of this element cannot be traced back to a repeatable formula. Through a combination of the development cycle's element iteration, (cycle B), and overall module assessment, (cycle A), the functionality of the illustration evolved into an interactive component supporting measure selection. Fine determination of the sequence of events which guided this growth in functionality as a guide to future element development was not attempted. Further definition of the design sequence would only serve to constrain the wide range of alternatives open to the design of future elements. Design constraints placed on an essentially creative community of developers (Bott 1988, Wroblewski 1991) may account for the incomplete adoption of formal methodologies (Raghavan and Chand 1989, Robson et. al. 1991). The important feature of effective design for the illustrative graphic lay in the balance between support for creative design and the controlling inputs of the users and developer throughout the design process. Critical to the creative design and evolution of this element was the flexibility of the prototyping environment. A less flexible prototyping/development tool may have impeded this line of design exploration.

The specification of the module passed through several forms. It advanced from the views of the group members concerning the information content to an agreed functional definition. The developer translated this into a representative design in the form of a non-functional prototype, via a list of prescribed functions. This prototype became the specification document. By illustrating the concept, the prototype prompted discussion and provided the means for representing the conclusions reached by the discussion. The resulting design issues were effectively explored by following the prototype development cycle.

The prototype development cycle's cycle A was followed to produce the first non-functional interface. The act of laying out the list functions within the interface window forced the consideration of the relationship between elements and highlighted the need for additional elements, continuing cycle A's top level review. User feedback led to a series of small design changes at this top level.

The group's broad acceptance of the developer's design approach led to further development of the non-functional components the prototype. The group members did not have the time or interest in the detailed design of each of these components to
provide feedback during their design. It was the developer who continued the prototype development cycle by selecting and developing each component. The design of each code or graphic component was based on an iterative cycle of design and testing centring on the developer, (cycle B). This included a sub-cycle, (cycle C), representing the iterative nature of design for each component of the prototype. Acceptance of an element’s design would lead back to an assessment of the prototype which would prompt either a top level change to the prototype or the selection of another element for further design. It was at this central point of prototype evaluation that user feedback was both sought and offered.

Whilst the prototype development cycle offers an apparently sequential activity it should be noted that user input to the design process was unrestricted and untimed. This was a response to the in-house development environment in which the group members were continually exposed to the evolving design. No attempt was made to constrain the form of their feedback. Comments relating to overall design would lead to gross changes to the design based on cycle A. Comments or observed problems relating to specific components would lead to their redesign based on the developer centred cycles B and C. Whilst these cycles are designated as developer centred, users occasionally contributed to the design of elements.

The emphasis of design effort remained with the developer. Whilst the users had a stake in the product, the input to the development project fell outside their specified work programme. User input to the project was considered to be unaccountable time. This stance reduced the hidden development cost of lost accountable user time. In doing so, it increased the need for good working relations between the developer and the user group to maintain consistent user involvement. The achievement of these conditions owes more to organisational culture and individual personalities within the group than to procedures to foster user involvement. Whilst the users were pleased to help the design process they were anxious not to offend people within the group, including the developer. In order to gain the required feedback within the prototype development cycle, the developer had to actively encourage constructive criticism.
6.7.3 Module Acceptance

As discussed earlier, the joint purpose of the lifecycle's acceptance phase was firstly, to confirm the module's suitability for the specified function, and secondly, to highlight possible improvements to the design. The lifecycle's allowance for design iteration made the tasks of the acceptance phase an extension of the prototype development cycle. The distinction between the acceptance phase and cycle A of the prototype development cycle was the acceptance phase's use of formal evaluation procedures.

Howard and Murray (1987) set out five approaches to evaluation, (section 3.6.4). Whilst, theory based evaluation was excluded due to the absence of a user model as a starting point, each of the other approaches was incorporated in the module's development. User based evaluation through informal feedback formed a key part of the prototype development cycle. Both expert based evaluation and subject based evaluation were applied following design evolution within the lifecycle's acceptance phase. In addition, a form of market based evaluation was applied to the continued evaluation of the Anthropometry module following delivery, leading to a subsequent upgrade of the module, (section 9.4).

Assessment of the anthropometry module was undertaken at two levels. The first was the informal assessment of the evolving module by both the developer and the members of the user group. It was this level of assessment which guided product evolution through the prototype development cycle. The second level of assessment was formal evaluation and reporting as part of the lifecycle's acceptance phase. These two approaches are characterised below.

**Informal**
1. Regular user responses to the evolving prototype design.
2. Developer assessment and iteration of all aspects of design.

**Formal**
3. Expert evaluation of the prototype by an impartial individual.
4. User Evaluation through controlled user testing.

The informal prototype assessments were discussed earlier. The following sections consider the application of the chosen formal techniques within the acceptance phase.
6.7.3.1 Expert Evaluation

It was considered that an external expert could bring the following beneficial attributes to the evaluation.

- An alternative perspective to highlight design errors overlooked due to the developer’s continual close contact with the product.

- An impartial opinion on the suitability of the design and the development approach for the product under evaluation.

- Prior product development experience to predict potential problems and suggest possible solutions.

- A knowledge of the software design guidelines.

In practice, expert evaluation provided suggestions for functional enhancements rather than highlighting design errors. Several recommendations were rejected through the discussion process, due to the unavailability of the suggested additions to the data set. Whilst the few recommendations implemented in the iteration of the Anthropometry module had a minor effect on the overall application, they required little extra development effort due to the flexibility of the prototyping tool.

The intended role of the expert evaluation process was a safeguard against the inclusion of significant design problems. The relatively low number of recommendations produced by this process may be due to either a failure of the process or the success of the prototyping cycle to develop an appropriate module. The success of this process can be assessed against two criteria. The number of design issues raised and the number of design issues missed. The low number and importance of the issues raised suggests a failure of the approach. This could be attributed to inappropriate expert selection or the evaluation and discussion approach. However, consideration of the results of user evaluation and post delivery maintenance suggests that the prototype did not contain critical design problems. The lifecycle might be seen to have failed if significant design problems were encountered at a later point. The explanation for the small contribution of expert evaluation was considered to relate to its position in the lifecycle. Important design issues might have been resolved more efficiently if the expert had assessed an earlier form of the prototype, altering its role from a pre-delivery safeguard to a prototype design technique.
6.7.3.2 User Evaluation

Expert evaluation was applied as a safeguard against inappropriate design. The expert's experience helped to assess the prototype for the contravention of 'good design practice'. However from a user centred design perspective, the most important issue in the assessment of the product's suitability was its acceptability to the defined user population. An indication of user acceptance for the product was achieved informally through the subjective responses of users within the prototype development cycle. The acceptance phase sought to confirm the product's fitness for purpose through objective measurement of the users' ability to access the functions of the product.

There were two main aims of formal user evaluation. The first was to judge the ability of each member of the group to accurately use the tool for anthropometric data retrieval. This was supported by the results of a set test to be completed with the assistance of the prototype tool. The second aim was to reveal possible design improvements. This was supported by an analysis of the background completion logs and by user feedback.

The user test results were effective in determining two cases of inappropriate design, printing and finding measurement details. In addition, the test results confirmed the ability of the group to access the remaining functions. This allowed the module to be accepted on the condition of correcting the two design problems.

The use of background logging provided a method of considering the users task sequence whilst retaining a realistic setting for module interaction. The subjects completed the task at their own convenience, at their own pace, in the actual environment for the planned implementation of the system. This provided a more representative testing environment than the regular usability laboratory approaches of direct observation, video recording or post task interview (Nielsen 1994). Background logging of user task completion without observation goes beyond the approach adopted by Palmiter (1994) of observing the user within their actual working environment for achieving an ecologically valid evaluation (Macleod 1996). However, a number of issues countered the advantages of background logging. The main issue was the loss of experimental control associated with the absence of an observer. In addition, the presence of an observer would have allowed the record of user strategies and emotional responses to
the system such as frustration or satisfaction. Although the background log provided a comprehensive list of completed actions, the time required to analyse and interpret the text based action log was much greater than the total task completion time of the users. In view of the limited development resources, the added analysis time was not considered to be justified by the advantages background logging.

Despite its rejection for formal user evaluation, the background logging component of the module function list was implemented in subsequent module designs. The low overall development cost of its continued inclusion and the future possibility of reducing the analysis task through an automatic summary of the log data led to its retention.

6.7.4 Delivery and Maintenance

The inclusion of an iterative cycle within the delivery phase proved unnecessary for the Anthropometry module. This indicated that all significant design faults had been successfully addressed through the iterative processes within the development and acceptance phases. Alternatively, design problems may have remained unnoticed within the design due to a low overall level of usage, (chapter 10). More intensive use of the tool by a larger user base may have revealed design problems missed by the lifecycle's assessment procedures. However, as problems did not appear over the longer term, this possibility can be rejected.

The apparent absence of faults in the delivered product was a positive finding. A negative view is that this was the result of over-development. Although avoidance of faults in the delivered product was the project's goal, the absence of minor design corrections suggests that the prototype may have reached an acceptable standard for delivery earlier in the lifecycle. If this was the case, the efficiency of the cycle would have been improved by earlier delivery.

The risk of over development was not recognised at the start of the Anthropometry module project. The strength of the prototype development cycle lay in its ability to improve the initial design through evolution. However, the cycle did not include a formal end point at which the prototype must be passed to the acceptance phase. This problem was compounded by the perceived importance of successful delivery of the system's first module which was considered to delay its formal release to the users for
evaluation. By recognising this risk, the development cycle of the subsequent module attempts to address this problem.

The approach to module maintenance cannot be considered over the short term as it was not required. This is not to say that it wasn't required over the longer term. Section 9.4 describes the changes made to the Anthropometry module later in the project as a response to emerging requirements. The absence of an initial maintenance activity was attributed to a combination of the module's simplicity and its over-development. The task of maintenance was included in the lifecycle model as a simple iterative cycle after delivery. Whilst this was not required for the Anthropometry module, it was retained in anticipation of the need to modify subsequent modules immediately after release due to increased complexity or earlier delivery.

6.8 Lifecycle Redesign for a Modular System

The development of the Anthropometry module followed the sequential phases set out in figure 6.1. The Anthropometry module case study allowed the refinement of this linear development process including the addition of a detailed specification phase and the selection of techniques to support the remaining lifecycle phases. The product for the study was a more detailed linear model for application development by prototype evolution. However, this linear model did not account for the view of the DAVE system as a collection of centrally linked modules. Instead it considered each module to be a separate application.

Figure 6.9 presents a redesign of the lifecycle model to match the design process of a modular system. This 'modular system' lifecycle separated the process into two levels. The first was the system specification process in which the group's requirements were analysed, the platform and tools chosen and the system's content loosely specified. 'System level' specification was followed by the sequential framework determined by the development of the Anthropometry module. This 'module level' sequence started with the more detailed 'module' specification followed by design evolution through the prototype development cycle, formal acceptance and finally delivery. The model then returned to a review of the system as a whole to select the next area for development.
The traditional linear model placed maintenance as the final, on-going task of the development project. This does not fit a scenario in which resources are limited to a single developer. The revised model separated the maintenance function into two tasks. The first was the immediate correction of faults after delivery. This was placed within the delivery phase of module development. This sequence was suggested as a limited period after which the project's work programme should be moved on to system re-specification and the start of a new module's development sequence. In order to manage requests for maintenance of a delivered product, the revised lifecycle model provides an additional maintenance phase. The 'modular system' lifecycle provided an option for 'system level' specification to divert development resources to the maintenance of a delivered product before the start of a new development sequence. Chapter 9 considers the role of this separate process.

Figure 6.9 Lifecycle for the Evolution of a Modular System
6.9 Conclusions

Development of the Anthropometry module provided a case study in which an evolutionary prototyping approach was applied to the design and development of a product starting with a loose concept. The results of the formal expert and user evaluations confirmed the product's achievement of acceptable levels of functionality and usability. The chosen development process sequence led to the delivery of a product which reached and exceeded its specification. The suitability of the specification of both the information area and the module details depends on a longitudinal view of module acceptance. Success of the product is considered further in chapter 10, based on its use over the full time-frame of the DAVE development project and its associated development cost.

The conclusions drawn at this point in the DAVE development project are presented below:

1. Prototype development from a minimum specification required a more detailed 'module level' specification phase as a starting point for design.
2. The definition of requirements appropriately involved the users by focusing the specification on how the information could be used rather than how the tool should provide the information.
3. Semi-structured interviews provided the necessary flexibility for requirements definition whilst ensuring consistent discussion across the group.
4. Construction of a function list provided an effective developer-centred technique to support the initial design of a non-functional prototype interface.
5. Informal user responses were not available until the product concept had been translated from its text-based specification documents into a tangible prototype.
6. Prototype evolution produced an acceptable design based on user feedback within the prototype development cycle.
7. The prototype development cycle lacked the mechanism to control the point at which the development phase should transfer to the acceptance phase, risking over-development.
8. Expert evaluation provided a minor benefit as an acceptance phase task, out-weighted by the associated effort.
9. User testing captured all remaining critical design iteration requirements.
10. Background logging provided sufficient information to evaluate the module but required too much analysis effort to justify this approach over direct observation.

11. The lack of demand for maintenance after delivery was due to both product simplicity and over-development.

Recommendations for the lifecycle model to guide the continued development of the DAVE system include:

1. Delivery of a module should be followed by review of the system and selection of a new module for development.
2. Module maintenance should be provided as an alternative to development when specifying the project's continued work programme.
3. Expert evaluation should be undertaken at an earlier point in the development lifecycle.
4. Transfer from the development cycle to the acceptance phase should be accelerated to avoid over development.
5. Background logging should be replaced with direct observation for formal user testing.
7 The Standards Module

7.1 Chapter Overview

Development of the first module prompted changes in the project's lifecycle model. Chapter seven presents the application of this amended model to the development of the second module of the DAVE system. It discusses the evolution of a loose concept into a deliverable product which provided the users with on-line retrieval and advanced searching of the group’s key reference texts.

The guiding lifecycle model for the second DAVE system module is outlined. The main changes to the previously applied development cycle are highlighted. Completion of the first module was followed by a return to the 'system level' specification phase. The processes of system review and product specification leading to the selection of the second module’s work programme are discussed.

A process model for the lifecycle's 'module level' specification phase is presented based on data set selection as a starting point. The phase's components are illustrated by the specification of the module. The role of the previously developed Text application as a design model is considered. Anticipated competition between media drove the iterative exploration of the functional advantages offered by the computer over paper based searches. The module’s solution of a 'free-text' indexing approach is introduced.

Expert evaluation was applied at the earlier stage of prototype development. The results of this evaluation are presented and the consequent design changes illustrated. The formal evaluation of this revised design through user testing was undertaken within the lifecycle’s acceptance phase. The evaluation results are presented and the alternative approach of direct observation is discussed. The interface of the delivered module is presented to illustrate the changes made following expert and user evaluation. The added task of product distribution is reported following the move to provide the DAVE system over a local area network.

The chapter's discussion of methods contrasts the lifecycle processes of the second module with the reported processes applied to the first module’s development cycle. The raised level of group experience led to a more complete specification of the product at
the 'system level' review stage. The role of the 'system level' specification phase is considered further and a revised process model is presented. This model separates the original background analysis phase from the lifecycle's repeated overview and formal specification activities. The processes of 'module level' specification are discussed, including the influence of a more detailed 'system level' specification, the available data sets and the existence of a suitable design model. The lifecycle's prototyping phase is reviewed and a revised prototype development cycle model is presented. The model further defines the levels of iterative design down to individual elements and identifies the point at which user and expert feedback provides an appropriate input to the prototype development cycle. The contribution of expert evaluation, as a source of development feedback within this cycle, is compared against it's role as an acceptance phase process. The use of direct observation in place of background logging in support of formal user testing is presented as a more appropriate approach for the given development scenario.

7.2 Lifecycle Approach

7.2.1 Model Overview

The early indications of the suitability of the Anthropometry module's design, such as the usability evaluation results and user response, supported the continued use of the evolutionary prototyping approach laid out by the lifecycle methodology, (figure 6.9). Figure 7.1 separates its framework into the three segments of System Specification, Module Development and Module Maintenance. Development of the DAVE system's second module followed the same lifecycle structure whilst revising its component activities based on the recommendations from the first development project. Changes made included the incorporation of expert evaluation within the prototyping phase and the stated intention of early transfer from the development cycle to the formal acceptance phase.
7.2.2 Specification and Development

The 'modular system' lifecycle directed that completion of the Anthropometry module should be followed by a reassessment of the complete product in a return to the 'system level' specification phase. The processes chosen for the initial 'system level' specification were primarily dictated by gaps in the background information, (chapter 4). The information required related to the group's work programme, user expectations, organisational environment, suitable tools, etc. With these gaps already filled, the return to the 'system level' specification phase held a different purpose. Instead, it was considered necessary to reassess the system in terms of its fitness for its intended purpose prior to the specification of the continued work programme.

User group discussion in the original specification phase was effective in directing the area of development for the first module without constraining the subsequent design process. Consequently, formal user discussion was retained for the continued specification of the project's work programme. The 'modular system' lifecycle, (figure 7.1), offered the option of either undertaking the development of the next module or
carrying out maintenance on existing modules. By separating the development and maintenance activities, the study recognised the project's specific resource constraints. Whilst a managed software development team might have been able to tackle these phases in parallel, the DAVE project was limited to one person to carry out the work. Although this does not prevent parallel activities, dividing the tasks into separate sequential tasks assisted project management at this level. At this time, no evidence had been revealed for the need to undertake DAVE system maintenance. The direction chosen through this 'system level' specification was the development of a second module.

7.2.3 Evaluation

Expert evaluation, (Howard and Murray 1987), for the first module was considered to be ineffective due to the prototype's advanced state of design, (section 6.7.3.1). For the second module, it was proposed that expert evaluation should take place at a point where the prototype represented the full functionality of the module in its interface but before the system was made robust. In this new position, the aim of the expert evaluation task became to guide continued prototype evolution rather than simply approve the design prior to delivery.

The second concern raised by the development of the Anthropometry module was the selection of the point of transfer to the acceptance phase as this could not be adequately predetermined. The approach taken was to adopt the design principle of early transfer, i.e. initiate formal evaluation before comprehensively iterating the design at a developer level. This approach was favoured over a rigid criteria for transfer. Such a measure was considered to be unworkable due to the wide variation in design alternatives. The expected consequence was a higher number of problems raised by formal evaluation. The considered benefit was an assurance that development effort was only expended on those design issues considered important by the user and the expert evaluator, in conjunction with the developer.

Moving the expert evaluation process to the prototype development cycle reduced the acceptance phase to the processes of formal user evaluation followed by design iteration. User evaluation became the sole formal quality control process prescribed prior to delivery. Another change was the adoption of direct observation of task completion followed by user interview in preference to background logging. Background logging was
retained as a potentially useful data source. However, its data was not intended for formal module evaluation within the development lifecycle or as a user performance measure, addressing the concerns of Irving, Higgins and Safayeni (1986).

7.2.4 Delivery

The components of the delivery phase remained unaltered from the previously applied lifecycle model. Distribution was not seen to be a requirement as the second module was to be implemented on the standalone platform along side the Anthropometry module. The iterative fault correction process was retained as part of the delivery phase in anticipation of a higher fault level for the second module, linked to the proposed tightening of the delivery time-scale.

Completion of the second module's delivery phase was to be followed by a return to 'system level' specification. From this point the project's work programme could be directed towards either the development of a third module, or the maintenance of existing modules. The absence of the requirement for maintenance up to this point removed information on which to base the definition of suitable maintenance processes.

7.3 System Specification

7.3.1 System Review

User involvement in system review was limited to informal feedback. The close working environment ensured that this source provided a continual monitor of the group's view of the DAVE system and how it related to their evolving working environment. A formal review of these factors was considered to be unnecessary. This was considered to be an inappropriate diversion of the group's efforts from their regular work programme as this information was continually available through informal user feedback as a consequence of the working environment. This user effort was not formally accounted for and generally occurred in the spare-time of individuals, during the organisation's regular break periods.
No major changes in the organisational environment or user group's composition had taken place following the previous completion the system specification phase. Equally, no requirements changes relating to the delivered module were revealed by the informal survey of the group members. Feedback concerning the delivered system was favourable. It did not raise maintenance tasks to be considered in place of new module development. This allowed the project to move onto the specification of the next module.

A new computer platform had been acquired to serve as a separate development platform, releasing the original computer for the standalone presentation of the implemented DAVE system. This allowed group's members to gain unrestricted access to the system without altering the chosen approach to system presentation.

The system assessment process acted to inform both the developer and users prior to the discussion and specification of the project's continued work programme. The activity of group discussion was found to be appropriate for the specification of the first module's development work programme. This led to the selection of formal group discussion as the preferred method for continued work programme specification.

7.3.2 Work Programme Specification

Formal group discussion for the original specification of the development work programme was preceded by the distribution of a list containing possible development subject areas to all members of the group prior to the meeting. The original work programme specification document, (table 4.5), was reused as the agenda for discussion. As all group members were involved in the prioritisation of the work programme, each understood the document's content and held a stake in any changes made to it.

The absence of a maintenance requirement allowed the meeting to concentrate on selecting a product for development. The original formal group meeting centred on the discussion of subject areas as directed by the developer. The second meeting started with a discussion of information areas, but expanded into a discussion of the next module's functionality based on the group's increased understanding of the scope and limitations of the development task by the group's members.
The chosen development area was text retrieval of the group's key reference documents. The group discussion went on to consider the appropriate text sources for the system and approaches to their acquisition. The group's opinion was divided as to the suitability and priority of possible text document sources for inclusion. The developer's preference for the concentration on internally produced documents was overruled by the group in preference for assisted access to broader reference texts. The meeting concluded by directing the development of a text indexing and retrieval software tool. The database was to contain design standards documents regularly referred to by the group. Further text subject areas were left for consideration later in the project.

7.4 Module Specification

Specification of the Standards module adopted the methods first applied to the Anthropometry module. These procedures included data set selection, requirements definition and the construction of a function list.

'System level' specification of the first module identified anthropometry as its required data set. The first activity of its 'module level' specification was the selection of this data source. This data set was further defined following requirements definition and function list construction. In the same way, the group identified text based reference documents as the required data to be supported by the second module. The documents considered to be necessary for the module's data set were selected. As noted for the Anthropometry module, (section 6.3.1), the form of the selected data imposed requirements on the module's functionality. Further refinement of the data set was not required as the decision was taken to include the full text of all selected reference documents. Function list construction was undertaken as the last specification process. The applied sequence of events for the specification of the Standards module is illustrated in figure 7.2.
7.4.1 Data Set Selection

Data set selection for the Anthropometry module identified a large single source of data and filtered it for relevant measures. Data set selection for the Standards module required selection from a wider range of alternative data sources. In addition, each of these sources could be reduced to relevant chapters, paragraphs or quotations.

The users' specification of 'Standards' references within the group meeting was confirmed and refined through user interview. The reference documents considered to be specifically relevant to the work of the group are listed in table 7.1.
Table 7.1 VDS Reference Documents

Omitted from this list were internal reports in which the research experience of the group was documented. The large number of past reports and their low frequency of use made them inappropriate for inclusion in the specified data set.

The reference document list is shown in order of perceived relevance by user group members. The users were asked to indicate the relative importance of the list text sources based on their usage of each document. Defence Standard 00-25 (00-25) was the prescribed military human factors standard for the British Army. The group considered it to be their primary reference source. The American Military Standard 1472D (1472D) was thought to be an important secondary source of relevant information. The remaining documents were only used occasionally and considered to be less specifically related to the work of the group. 00-25 was specified as the primary source of data for the proposed system. Acquisition of the remaining documents in the list was scheduled for consideration following completion of the 00-25 database. The users considered all sections of the two specified standards to be relevant to the work of the group.
7.4.2 Requirements Definition

The basic functionality of the module was set during the formal meeting. This could be seen to fulfil the purpose of the requirements definition process of the module specification phase. However, the definition of requirements was retained to formalise the functional specification.

Semi-structured interview of all user group members revealed a diverse use of the reference texts ranging from references for reports to answering telephone enquiries. When applied for the previous module, the requirements definition took the form of model enquiries. The range and diversity of possible texts and their application prevented the determination of representative model questions. Instead, a formal statement of functional requirement was produced that was acceptable to the group members. The required function of the proposed module was stated to be:

- To provide enhanced access to the group's frequently referenced documents.

Hidden behind this basic requirement definition lay inherent requirements for module success. Unlike the first module, the proposed tool would be required to complete on equal terms with an existing information source. Consequently the module needed to provide a sufficient advantage over the paper source to motivate usage. How to provide this advantage was not clear at this point in the process. The user group looked to the developer to explore solutions to this problem.

7.4.3 Function List Construction

Table 7.2 presents the function list constructed as the starting point for the prototype development cycle. It includes the elements considered necessary to display and retrieve the text with additional elements considered to utilise the functional advantages of computer presentation. The list includes standard elements of the currently implemented DAVE system.
Application Function

<table>
<thead>
<tr>
<th></th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Retrieve text from File</td>
</tr>
<tr>
<td>2</td>
<td>Display reference document title</td>
</tr>
<tr>
<td>3</td>
<td>Display reference text</td>
</tr>
<tr>
<td>4</td>
<td>Navigate through reference text</td>
</tr>
<tr>
<td>5</td>
<td>Search text</td>
</tr>
<tr>
<td>6</td>
<td>Record user notes</td>
</tr>
<tr>
<td>7</td>
<td>Print</td>
</tr>
<tr>
<td>8</td>
<td>Display document size</td>
</tr>
<tr>
<td>9</td>
<td>Show current location within the document</td>
</tr>
<tr>
<td>10</td>
<td>Display module's title</td>
</tr>
<tr>
<td>11</td>
<td>Return to entry screen</td>
</tr>
<tr>
<td>12</td>
<td>Log events</td>
</tr>
</tbody>
</table>

Table 7.2 Standards Module Function List

7.5 Prototype Development Cycle

7.5.1 Document Acquisition

As noted for the development of the first module, there was an initial split in development effort between prototype design and data set acquisition. The technique of optical character recognition (OCR) had improved sufficiently since the start of the project to handle the 00-25 document with a greater than 95% translation accuracy. OCR produced a file of recognised text including basic formatting. Scanning and character recognition was followed by manual error correction with the aid of a computer based spelling checker. The final data set for 00-25 consisted of 623 pages, divided into the document's 12 parts.

7.5.2 A Design Model

The development of the test applications, (section 5.3), were reported to contribute to the design of the DAVE system's entry screen and first module in addition to confirming
the scope of the development environment. Development of the Text application, (section 5.3.2), went further by acting as an early prototype for the Standards module.

The basic mechanisms of text retrieval and presentation had been explored through development and use of this test application. It confirmed the ability of the development environment to handle large volumes of text. At the time of its development the facilities for document scanning and OCR were not available to the development project. The tools for document acquisition became available during the development period of the DAVE entry screen and the first module. The Standards module's specification process highlighted text searching as the main advantage offered by computer presentation of text documents. This advantage had been demonstrated using the test application's simple text string search.

7.5.3 Network Distribution

The Standards module project coincided with the acquisition of several desktop computers, compatible with the DAVE system software. This raised the possibility of presenting the tool on each of the main users' desks as an 'on-line' information tool. Whilst an increase in the group's level of computer usage was anticipated, uncertainties concerning platform dominance within the organisation and networking opportunities blocked the development of the DAVE system as a network tool from the start. The network option was made possible by the developer setting up a local area network within the group's offices.

A consequence of pursuing a network approach to the DAVE system was the need to adopt the smaller screen size of the recently acquired computers. Presentation of the DAVE system on the new computers required a change in the project's interface standard. The Anthropometry module had been developed for presentation on a 19 inch colour monitor. Redesign of the Anthropometry module to support the new 14 inch interface format was undertaken after the delivery of the Standards module, within the lifecycle's maintenance phase, (section 9.4).
7.5.4 The 'Free Text' Indexing Approach

The early Standards module prototypes continued the Text application's 'paper metaphor', presenting one page at a time. As noted for the initial design of the DAVE system's entry screen module, (section 5.3.2), an attempt was made to relate the design of the interface to familiar objects within the group's working environment. The user was able to navigate up and down a virtual stack of the document's pages using buttons. Whist the functionality presented by this test application might have been considered to meet the user specified aims for the system, it was not considered to provide a sufficient advantage over the paper source to promote continued usage. Additional text manipulation techniques were explored through the prototype development cycle with the goal of enhancing the tool's functionality. The approaches used included basic hypertext linking, indexing and text searches.

The preferred approach to document searching at this point of design was to mimic the indexing conventions recognised by the users in the form of author compiled tables. However, the level of post-acquisition document editing required for effective document indexing was considered to be impractical within the scope of the project. The more advanced document navigation technique of hypertext linking was rejected at this stage for the same reason. Unrestricted text searches were acceptable from a development perspective as they did not require further document editing for their use. The concern with this approach to document searching was that it transferred the search effort to the user. Unrestricted searching required the user to select suitable search keywords. This could be viewed in two ways. It was either a burden on the users, or it granted navigational freedom. The usability problem lay in the requirement of the user to guess which words might be used by the authors of the document. An incorrect guess could lead to the mistaken assumption that a topic was not covered by the searched document.

Iterative prototype development was used to explore a 'free text' approach in which every word in the acquired document was referenced. This approach was used to produce a collapsed list displaying every possible keyword and the number of occurrences allowing the user to visually search for appropriate keywords. User selection of a listed keyword prompted the system to display each occurrence in the context of the sentence in which it appears. Selection of a promising sentence would change the display to present the sentence in the context of the page in which it was found.
The approach to module help was altered to incorporate the new Macintosh operating system standard of 'balloon' help. When help was activated a context sensitive information balloon would appear when the cursor was positioned above a functional interface element. Construction of a 'balloon' help system represented less development effort than the animation approach incorporated in the first module. User acceptance of the new help system led to the later update of the Anthropometry module to include 'balloon' help.

7.5.5 Expert Evaluation

7.5.5.1 Approach

In line with the proposed alteration of the lifecycle procedure, (section 6.9), expert evaluation was undertaken within the prototype development cycle. Whilst the prototype's interface represented the full functionality proposed for the module, the development of the functional components was only partial.

Although undertaken at an earlier point in the development cycle, expert evaluation adopted the procedure followed for reviewing the first module. The same individual was asked to evaluate the second module. The functions of the prototype were demonstrated by the developer. The expert was then allowed to conduct a self-paced 'walk-through' evaluation of the prototype. Conclusions drawn by the expert were then discussed with the developer and a proposed iteration list drawn up.

7.5.5.2 Results

Conclusions drawn by the expert following evaluation are presented below.

- The quantity of text made the interface overwhelming.
- The functionality of some interface components was unclear.
- The overall interface was more complicated than the Anthropometry module.
- Scroll bars would offer a more standard approach to field scrolling.
- The functionality of the notes field could be limiting.
- The change to 'balloon' Help should be advertised.
The expert considered the overall complexity of the prototype to make the system inaccessible to less experienced members of the group. The central recommendation of the evaluation was the exploration of simplified approaches to presenting the information.

The early application of expert evaluation broadly indicated necessary areas of improvement. It concluded that the prototype would be unsuitable for implementation without offering specific solutions for improvement. It recommended a shift in design emphasis from indexing functionality to interface simplicity. These comments were accepted by the developer and provided a focus for continued development.

7.5.6 Prototype Redesign

The results of continued prototype evolution following expert evaluation are illustrated in figures 7.3 and 7.4. Figure 7.3 presents the interface of the prototype's Index screen. This design offered a simplified interface. The 'notes' window was removed which allowed the 'index' and 'context' windows to be extended. The module could be used at three levels. By pressing the 'document' button the users could browse the selected document page by page. Alternatively, they could visually search for keywords in the word list of the module's Index screen. Finally, the more advanced features of word searching and Boolean filtering were provided by input fields at the bottom of the screen.

Figure 7.4 presents the interface of the prototype's Document screen. The interface presents a half page display with a scroll bar for viewing hidden sections of the selected page. The selected keyword was presented in the centre of the screen and highlighted. A 'copy' button grabbed the highlighted text which could then be pasted into other documents. The page number was presented at the bottom of the screen and indicated on an icon representing a stack of pages. The user could navigate through the document page by page or by entering a page number to 'jump' to.
The initial prototype workplace is built from the specifications of the workplace prototype. Modifications if necessary, allied design of the prototype workplace prior to its production.

12.1 Introduction

The purpose of evaluating the prototype workplace is to verify the operating and maintenance effectiveness of the workplace under actual operating conditions. The prototype workplace should be tested prior to committing it to production. Tests should include operation by the military users and maintainers who represent the final user population. Although initial tests may be made by specialists in order to ensure practicality and safety of operation, the key test is whether typical users can and will operate the workplace as planned. Quantitative measurement of human-machine performance should be carried out whenever the complexity or safety of the workplace is critical.

12.1.2 For mobile and land-based systems, early prototype trials (including user functional assessments), are normally carried out for the user by specialist Ministry of Defence trials groups or units. These units keep a running log of defects arising from prolonged operation of the workplace prototype. Often an employee from the contractor’s design team is assigned or on immediate call to the unit to repair or replace defective components as necessary. When defects are either frequent or require a major redesign, they also become the subject of task requests for consideration by MDD, before action is placed on the designer to introduce them into the next prototype rework.

12.1.3 Later prototype trials are usually carried out in the theatre of use and under quasi-operational conditions. For a mobile and land-based system, its interaction as a new sub-unit within the larger organization.
The inclusion of a 'camera' button represented an important addition to the functionality of the module. Informal user and developer interaction with the text based prototype highlighted the practical importance of the graphical content of the specified documents. This additional feature allowed the user to replace the recognised plain text file with a photographic representation of the scanned page. The user could then zoom and scroll the enlarged image to read the full text and graphics of the document. These page images were produced as a by-product of the OCR process. Due to the size of this picture database these images were stored on a central server to be accessed by the delivered modules across the emerging network.

7.5.7 Accelerated Transfer

Figures 7.3 and 7.4 represent one version of an evolving prototype. This prototype passed through more than 20 distinct interface variations. The interfaces presented do not represent the developer's assured choice of the best design alternative. Rather, it represents a point at which the main functional concepts arising from the prototype development cycle's feedback processes were incorporated. Several design options were considered for further iteration of the prototype. The tendency to overdevelop, as noted for the previous module, led to the rejection of several minor iteration possibilities. In addition, transfer to the formal acceptance phase was undertaken before full completion of the required database.

7.6 Acceptance

Transfer of the expert evaluation process from the acceptance phase to the prototyping phase reduced the proposed formal acceptance processes to only user evaluation followed by the option of design iteration. As the last quality control procedure it was increasingly important that this procedure should successfully detect serious design faults prior to delivery.
7.6.1 Formal User Evaluation

7.6.1.1 Aims

The aims of the formal evaluation process for the second module closely follow those set out for the first, (section 6.5.2.1). A user centred approach had been taken to the specification and development of the first module. The aims of its evaluation were to test the group's ability to use the product to effectively meet the proposed aims of the tool, and to gain an indication of whether the product met their expectations. For the second module, accelerated transfer to the acceptance phase increased the importance of highlighting features requiring iteration.

7.6.1.2 Method

In contrast to the effective monopoly on information held by the first module, the Standards module was required to compete with the accessible paper source for usage. Consequently, the approach taken to evaluation was a direct comparison of the group's ability to use these alternative media to answer representative enquiries.

A telephone enquiry log generated by the group over the preceding year was used to identify six question categories. A question was generated for each category by an expert in the group's work area from outside of the designated user population. The questions were rated by the expert to be of equal difficulty.

Each subject was presented with the DAVE system, including a revised 14 inch entry screen with links to the Anthropometry module and the Standards prototype. The Anthropometry module's link took the form of a dummy screen as the functional application was only available, at this stage, on a 19 inch screen. Placed next to the DAVE system terminal was the reference document from which the source text and graphics had been acquired. The experimenter demonstrated the completion of two example questions using both the Standards module and the paper source.

Comparison of the two media in this evaluation was intentionally biased. All of the subjects had experience in searching the selected reference document as part of their regular work programme. The subject's occasional input to the prototype development
cycle provided only limited experience of free text searching. By setting up this 'worst case' approach for the testing of the prototype the aim was to reveal design requirements rather than training the users to work around design faults.

The subject was asked to complete the six questions listed on their task sheet in order, (appendix C). The order of questions was balanced to overcome the potential learning effect, (table 7.3). Each user answered three questions with the aid of the paper document. The other three questions were answered using the Standards module. The subject had to write a single line answer and reference the page number on which the information was located. The task was self paced with particular emphasis on accuracy, in line with their work programme’s performance criteria. The task instructions directed the user towards the 'balloon' help system if assistance was required. Each subject completed the task within office hours.

<table>
<thead>
<tr>
<th>Subject \ Question</th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
<th>Q5</th>
<th>Q6</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td>E</td>
<td>F</td>
</tr>
<tr>
<td>ii</td>
<td>C</td>
<td>A</td>
<td>B</td>
<td>F</td>
<td>D</td>
<td>E</td>
</tr>
<tr>
<td>iii</td>
<td>B</td>
<td>C</td>
<td>A</td>
<td>E</td>
<td>F</td>
<td>D</td>
</tr>
<tr>
<td>iv</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td>E</td>
<td>F</td>
</tr>
<tr>
<td>v</td>
<td>C</td>
<td>A</td>
<td>B</td>
<td>F</td>
<td>D</td>
<td>E</td>
</tr>
<tr>
<td>vi</td>
<td>B</td>
<td>C</td>
<td>A</td>
<td>E</td>
<td>F</td>
<td>D</td>
</tr>
</tbody>
</table>

**Bold = Paper**

**Plain = Computer**

**A to F = Order of presentation**

Table 7.3 User Evaluation Question matrix

Measures taken included error rate and completion time. Partial or incorrect answers were recorded as an error. Completion time was measured from the opening of the document to the writing of the answer for the paper based task and from the first key/mouse press to the writing of the answer for the computer based task. The
observation log was based on task action categories which were determined before user testing.

Following completion, each subject was interviewed. The users were asked to express opinions on the design and function of the prototype based on their experience of the evaluation task. In addition, they were asked to comment on issues raised through the observation of the session.

7.6.1.3 Results

The results are presented in table 7.4 and 7.5. They show considerable variation in the completion time between subjects and between questions for the two methods of enquiry. The computer access times for the younger subjects were either close to the paper access times or reduced. Notably, the paper access times were considerably quicker than the computer times for the two older members of the group. This was attributed to their intimate knowledge of the document and their lack of previous computer usage.

<table>
<thead>
<tr>
<th>Question /Subject</th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
<th>Q5</th>
<th>Q6</th>
<th>Total Paper</th>
<th>Total Computer</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>342</td>
<td>112b</td>
<td>81</td>
<td>211</td>
<td>132</td>
<td>68</td>
<td>535</td>
<td>411</td>
<td>946</td>
</tr>
<tr>
<td>ii</td>
<td>98</td>
<td>88</td>
<td>160</td>
<td>228</td>
<td>52</td>
<td>75</td>
<td>346</td>
<td>355</td>
<td>701</td>
</tr>
<tr>
<td>iii</td>
<td>77</td>
<td>104b</td>
<td>122</td>
<td>338</td>
<td>315b</td>
<td>383</td>
<td>303</td>
<td>1036</td>
<td>1339</td>
</tr>
<tr>
<td>iv</td>
<td>150</td>
<td>507</td>
<td>407</td>
<td>372</td>
<td>197</td>
<td>237</td>
<td>1064</td>
<td>806</td>
<td>1870</td>
</tr>
<tr>
<td>v</td>
<td>192</td>
<td>332</td>
<td>155</td>
<td>472</td>
<td>702a</td>
<td>282</td>
<td>679</td>
<td>1402</td>
<td>2081</td>
</tr>
<tr>
<td>vi</td>
<td>287</td>
<td>83b</td>
<td>151</td>
<td>109</td>
<td>131</td>
<td>263</td>
<td>521</td>
<td>503</td>
<td>1024</td>
</tr>
<tr>
<td>Total Paper</td>
<td>517</td>
<td>304</td>
<td>363</td>
<td>953</td>
<td>1030</td>
<td>782</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total Computer</td>
<td>629</td>
<td>922</td>
<td>713</td>
<td>787</td>
<td>499</td>
<td>526</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>1146</td>
<td>1226</td>
<td>1076</td>
<td>1730</td>
<td>1529</td>
<td>1308</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Bold** = Paper  \( ^a \) = error (Fail)

**Plain** = Computer  \( ^b \) = error (Inaccurate)

Table 7.4 Completion Times (s) and Errors for the User Evaluation
Table 7.5 Summary of Completion Times (s) and Errors for the User Evaluation

As it was not possible for each subject to answer every question using both media, care must be taken in interpreting these results. As shown by the question matrix, subjects i to iii completed questions 1 to 3 using paper and the remainder using the prototype. This order was reversed for subjects 4 to 6. Any difference in performance for individual questions could be attributed to either the suitability of the media or to the aptitude and experience of the individuals. However, the individual performance results show a good balance between subjects i to iii and subjects iv to vi allowing us to view these results as an indication of media suitability. The subjects tended to answer the first three questions more quickly using the paper source and answer the last three more quickly using the prototype. The attribute of the questions considered to have caused this result was their categorisation. Questions 1 to 3 were easily located using a combination of the document’s section index and page flipping. In contrast, questions 4 to 6 did not fit clearly within the document’s section structure increasing the value of a free text search approach.

As the group’s requirement was quality over speed, the key test results were the error rates for the two media. Three errors were recorded for answers based on a paper search. Two of the errors related to insufficient detail in the answer. The third related to a failure to find the page reference. Two errors were recorded for questions completed by computer searching. Both computer based errors related to insufficient detail in the answer. Whilst the more serious error of recording an incorrect answer was not made, there was scope for improvement in referencing efficiency for both media. The single occurrence of failure to find a reference using the paper source was followed up by successful source location by the subject using the Standards prototype at the end of the test.
The low number of specified users, and hence evaluation subjects, prevent conclusions being drawn for the prototype's suitability for a wider population. The aims of the project did not require, or allow for, a wider evaluation. Neither was this considered to support the lifecycle goal of tailoring the tool to the chosen population. The subjects showed a wide spread in their performance for both media, demonstrating the loose nature of the task for which the computer tool was designed. The completion times show a slightly lower average time for answers using the paper source. However, the minimum completion time achieved and the range of times was lower for the computer tool.

The task action log mirrored the completion time results indicating a broadly similar number of actions, based on defined categories, for the two media. The results of this log are presented in tables 7.6 and 7.7.

<table>
<thead>
<tr>
<th>Subject \ Question</th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
<th>Q5</th>
<th>Q6</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>2 2</td>
<td>10B</td>
<td>8</td>
<td>9</td>
<td>8</td>
<td>5</td>
</tr>
<tr>
<td>ii</td>
<td>5</td>
<td>5</td>
<td>7</td>
<td>15</td>
<td>8</td>
<td>7</td>
</tr>
<tr>
<td>iii</td>
<td>7</td>
<td>7B</td>
<td>9</td>
<td>33</td>
<td>18B</td>
<td>30</td>
</tr>
<tr>
<td>iv</td>
<td>7</td>
<td>11</td>
<td>16</td>
<td>9</td>
<td>11</td>
<td>9</td>
</tr>
<tr>
<td>v</td>
<td>7</td>
<td>10</td>
<td>7</td>
<td>32</td>
<td>28A</td>
<td>19</td>
</tr>
<tr>
<td>vi</td>
<td>17</td>
<td>7B</td>
<td>8</td>
<td>5</td>
<td>8</td>
<td>19</td>
</tr>
</tbody>
</table>

**Bold** = Paper  
**A** = error (Fail)  
**Plain** = Computer  
**A** = error (Inaccurate)

Table 7.6 Task Actions and Errors for the User Evaluation

It was at the lower level of the task sequence that conclusions relevant to the development project could be drawn. Observation of the task showed two distinct search strategies. The computer tool demanded a structured approach in which a successful search was dependent the distillation of appropriate key words. In contrast, success using the paper source centred on visual skimming through the document. This approach hinged on the individual's previous experience of using the reference document. Experienced users of the reference document were occasionally able to locate the correct
section and immediately skim to the appropriate reference page of the paper source. However, failure by this method frequently resulted in random skimming through the document with little success.

<table>
<thead>
<tr>
<th></th>
<th>Total actions</th>
<th>Average actions</th>
<th>Min. actions</th>
<th>Max. actions</th>
<th>Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paper</td>
<td>220</td>
<td>12.22</td>
<td>5</td>
<td>32</td>
<td>(2b, 1a)</td>
</tr>
<tr>
<td>Computer</td>
<td>223</td>
<td>12.39</td>
<td>5</td>
<td>33</td>
<td>2b</td>
</tr>
</tbody>
</table>

**Bold** = Paper  
**Plain** = Computer

\(\alpha = \) error (Fail)  
\(\beta = \) error (Inaccurate)

Table 7.7 Summary of Task Actions and Errors for the User Evaluation

The similar performance of the two media under this ‘worst case’ approach was considered to support the acceptance of the computer tool, as there remained scope to improve both the design of tool and skills of the users. The important advantage of the computer based tool was seen to be the removal of the group’s reliance on user familiarity with the reference document.

Semi-structured interview following the test exercise was used to highlight both prototype specific and background issues. The interviews revealed an increase across the group in the use of computers as compared with the original background interview, (table 4.3). With one exception, the group used computers on a daily basis for word processing or data manipulation, (table 7.8). In contrast, use of the standards document was attributed to more variable work programme demands.
Table 7.8 The User Group's Computer and Reference Usage Profile.

The results of the user/developer discussion of prototype specific issues are summarised below.

User stated problems

- Would like to see more text in the context window. (subjects i, ii, iv, v)
- Would like the system to indicate when a photo was unavailable. (iv)
- Had trouble recognising whether the document was in text or photo mode. (vi)
- Had difficulty initiating a search of the index list. (v)
- Confused by the movement between the index and document screens. (iii, v)
- Would like to be able to flick between photos quickly. (vi)
- Would like the text to include cross-references to related texts. (vi)

User stated problems predominantly arose from confusion experienced when using the tool. They generally highlighted inappropriate features of interface design which could be solved by simple redesign of the interface features. Other issues raised, such as page flipping in 'Photo' mode, were restricted by hardware and software limitations.

The approach of direct observation, adopted for the formal user testing of the second module, allowed analysis of the design to move beyond the issues revealed by the data analysis approach applied to the evaluation of the first module. The observations made are summarised below.
Observed problems

- Use of index scrolling in place of faster index search function. (ii, iii, iv, v, vi)
- Occasional visual searching in place of faster Boolean filtering. (i, iv)
- Filtering of complete words rather than a wider, partial word, filter. (iv, vi)
- Unsuccessful use of document page flipping, mimicking paper skimming. (i)
- Non-computer user had initial confusion with scroll bars. (i)
- Failure to recognise the full functionality of scroll bars. (v)
- Double click to select a line when single was sufficient. (i, ii, iii, iv, v, vi)

The observed problems centred on user deviations from the optimal search strategy. Whilst this problem might have been avoided by pre-training, the design goal was to provide an intuitive interface to guide the user through the optimal strategy. The observed user search strategies raised possible design changes to bring the interface closer to its design goal. User training remained an option should difficulties persist.

7.6.2 Pre-Delivery Iteration

The recommendations for iteration drawn from the user evaluation results are listed below.

- Simplify document opening.
- Increase the effective viewing width of the context list.
- Improve the method of index searching.
- Improve text/photo mode indication.
- Improve index/document mode indication.
- Increase the document database.
- Fill all gaps in the photo database.
- Educate the users in the optimal search strategy.

Despite being effectively complete, the pre-implementation software was still considered to be a prototype due to the flexibility of the development tool. Although the project was advanced within its lifecycle, iterative alterations could still be made following the prototype development cycle. Changes made to the design of the prototype are detailed below.
1. Opening separate chapters was abandoned in favour of retrieving the complete document. This removed two actions previously required before a word search could begin. The cost of this approach was the increase in the index list length and the occurrence rate related to the increase in database size. Informal user testing showed this consequence to be acceptable. This concentration on two standards documents was accompanied by a change in the module’s name to the Standards module.

2. An increase in the context field width was achieved by adding the function of horizontal field scrolling. The cursor changed to an arrow, when moved to the side of the field, indicating the direction the list would scroll if the mouse key was pressed.

3. Index searching was advertised by the addition of a search button beside the input field.

4. User confusion in navigation between the two screens and other interface functions was addressed by the redesign of the related interface objects.

5. The photo files missing for the test were added to the database. Work continued to acquire the 1472D document.

6. The users were provided with simple background instruction concerning the most effective search approach as determined by analysis of the evaluation strategies used.

The completed modules as implemented are illustrated in figures 7.5 and 7.6.
Carried out at the prototype stage when a complete working model of evaluating the prototype workplace is to verify the operational feasibility of the hardware workplace prototype should be fabricated and evaluated.

Figure 7.5 Implemented Standards Module's Index Screen

Figure 7.6 Implemented Standards Module's Document Screen
7.7 Delivery

7.7.1 Distribution

The transfer of the DAVE system to a networked structure required the addition of a distribution activity to the delivery phase. The software was copied to the various desktop computers across the recently established network. The connection took the form of a local area network installed and maintained by the developer. This small network preceded the eventual networking of the full establishment, under the central control of the organisation. Throughout this period, the DAVE system continued to be delivered under the more stable local area network.

Included in the group's local network was a printer providing a common printing source. Previously, printing involved transfer of a document by disk to a computer directly linked to a printer. The convenience of the network link allowed distribution and configuration to be completed within a day. The convenience of this new network link contributed to the group's wider use of computers for document processing.

7.7.2 Post-Delivery Iteration

The increased complexity of this module, in conjunction with the wider distribution of the system, led to the revival of the initially proposed post-delivery iterative cycle, (figure 6.8). The main requirement for maintenance at the early stages following Standards module delivery was to match the software to the hardware configuration of the various networked computers. Beyond this, the software proved to be robust following delivery. It required little maintenance in the form of software correction. After this initial period of Standards module support the continuing general support process was carried out in parallel with the next phase of the development lifecycle, a return to 'system level' specification.
7.8 Discussion of Methods

As noted in the discussion of methods following completion of the first module, (section 6.7), the continued evaluation of the lifecycle depended primarily on an assessment of the completed product. Again the results of formal evaluation allowed conclusions to be drawn concerning the suitability of the delivered module and by association, the suitability of its lifecycle.

As concluded for the first module, the Standards module achieved its pre-defined aims, as confirmed by a ‘worst case’ evaluation. User centred prototype evolution within the ‘modular system’ lifecycle succeeded in guiding the design of a product found to be acceptable to the defined user population. Conclusions based on the objective measures of success of long-term usage levels and development cost are made in chapter 10.

7.8.1 ‘System Level’ Specification

The ‘system level’ specification phase provided the starting point of the original lifecycle model, (figure 4.1). It included the tasks of requirements analysis, product specification and tool selection, (figure 4.2). The evolution of the lifecycle model to support the cyclic concept of modular design and development led to the split in product specification tasks between the initial ‘system level’ specification and a new ‘module level’ specification phase concentrating on product details, (section 6.3).

As proposed by the ‘modular system’ lifecycle, (section 6.8), completion of the Anthropometry module was followed by a return to ‘system level’ specification. The aims of the original specification phase were to both determine the group’s detailed background and to integrate the developer within the group. The return to the ‘system level’ specification brought with it a detailed background knowledge, and situational information drawn from group integration over the period of the project. Whilst all of the detailed information determined earlier had a bearing on the specification of the continued work programme, the specification phase needed only to highlight changes in the group’s characteristics and requirements. Consequently the ‘system level’ specification processes took the form of an overview as opposed to the detailed analysis previously undertaken.
The original 'system level' specification phase culminated with the group's direction of the project's work programme based on loose information areas considered suitable for support. At this point the specification attempted to cover the complete development programme from the start. This compares with the traditional approach which sought to specify the system in full before development. Over the period of development user opinions revealed changes in the group's concepts of the future tool. This experience led the second formal group discussion's restriction of it's aims to the definition of the immediate work programme of the project.

The revised approach to the specification phase is illustrated in figure 7.7. The original detailed analysis of the organisational environment, including developer induction, is represented as a separate 'background analysis' activity within the specification phase. This is followed by an overview of the determined requirements and the current design of the product. This overview segment represented the point at which the current state of the requirements and design was assessed and related to topics for group discussion. Although not recognised at the time, a form of this developer centred overview process was completed for the previous 'system level' specification when identifying issues for group discussion. This model may therefore be applied to both initial system design and mid-life assessment.

The group's willingness to discuss these matters during their work breaks may have been influenced by the organisational culture. The group's management viewed work breaks as a valuable forum for peer group discussion. Whilst breaks were at regular times, for officially defined durations, discussion beyond the set times was common and unchallenged. A more rigid organisational stand point would have cut short these discussions about the DAVE system and may have made individuals less willing to discuss work related topics.

A risk faced by the project by relying on informal feedback at this stage lay in the influence of personal politics within the enforced confines of in-house development. Positive user response to the product might be based on a desire to avoid offence to a colleague. Alternatively, a clash of personalities could have led to the rejection of the product without full consideration. Despite these risks, system review remained a point of user contact with the development programme. This helped to include the users thus avoiding a possible cause of user alienation (Cooper 1994).
A notable feature of the return to the system specification phase was the increased understanding held by the group members of the scope of the development project. The range and detail of opinions held by the group members was raised, at the level of both informal feedback and formal group discussion. The increased maturity of user feedback was evident in the comparison of 'system level' specification for the first and the second modules. Specification of the first module produced a broad definition which was explored by the developer through iterative prototyping until an acceptable prototype was produced. In contrast, group discussion for the second module extended 'system level' specification to include proposed functionality.
'System level' specification through a formal group meeting as applied originally, (section 4.5), was retained as an appropriate method of specification. A previous limitation of this approach was observed to be a variation in user contributions to the discussion. For the second module, the level of contribution was more even across the remaining members of the group indicating an increasing confidence in respect to computer related issues. This broadened the range of input to include the older group members who whilst having more work related experience, were less computer literate and therefore reluctant to contribute to the original specification process.

The first formal group meeting aimed to prioritise the subject list as the basis for the project's complete work programme. This return to 'system level' specification revealed the long term work plan to be inappropriate. The proposed development task sequence was too limiting in view of the more mature expectations of both the user group and the developer. As a consequence, this second formal group meeting restricted its aims to the specification of the immediate work programme.

It is important to note that whilst this 'system level' specification was extended in comparison to the first development project, it continued to follow the principle of minimum critical specification (Cherns 1976). 'System level' specification at the start of the DAVE project did not extend to module functionality due to uncertainty in terms of how this information could or should be provided using a computer. Whilst the resulting specification was minimal it was also the maximum that the group was able to confidently prescribe. The group's growing experience provided confidence in extending and articulating their concepts for the second module enhancing the group's ability to be prescriptive. Whilst it was more detailed than previously, the 'system level' specification of the second module can also be considered to be minimal as it remained broad and flexible, avoiding unnecessary restrictions on the design process.

7.8.2 'Module Level' Specification

The more detailed 'system level' specification prompted the reassessment of the 'module level' specification approach for the second project. 'Module level' specification of the Anthropometry module, (section 6.3), followed data set selection, requirements definition and functional list creation with further refinement of the module's data set. Although the establishment's anthropometric database was recognised as the prime
source of data for this module at the start, its eventual reduction was dictated by the functional requirements of the group. As the resources of the project did not allow for further data gathering, the data set was confined to existing data.

The 'module level' specification of the Standards module recognised the data source as the defining factor from the start. The diversity in form and content of the reference documents suggested for inclusion made particular demands on the eventual functionality of the product. Whilst user requirements were considered to be the focus of the specification process, it was necessary to channel the definition of these requirements within the limits of the project's resources and hence the existing references available to the group. The approach taken was the initial selection of sources for inclusion followed by definition of requirements. It was then possible to propose prototype elements and undertake iterative design tailored to the form of the specified documents.

Selection of source references by user interview provided an acceptable approach. The available references were well known by the group and their relative importance to the group's work programme was well established. However, the use of interviews for the definition of user requirements was less successful. Little feedback was gained from the users concerning their functional requirements beyond those voiced at the 'system level' specification phase. Although their broader software development/specification experience allowed the users to feel confident in commenting on the general functional approach of the proposed product, when asked to define specific functional components they lacked the experience of other systems on which to base an opinion.

The result was a minimal specification which suited the prototyping approach adopted for the first module. The developer was able to explore functional design alternatives through the prototype development cycle. The absence of rigid user opinions concerning the functional design removed a potential constraint on the developer's design options. This allowed the developer to focus on the key issue of competing against the paper alternative for document referencing. Addressing this imperative within the design depended more on awareness of existing text retrieval techniques, creative design and prototype evolution than the set processes of the 'module level' specification phase.
Function list construction, as applied in the design of the first module, was used as a guide to the initial prototype design of the second module. However, the existence of the text retrieval test application pre-empted the functional list. Although a function list was produced, it only made a small additional contribution to the design process. The design adopted the Text application as the design starting point. The functional list added only the emerging DAVE standard components such as the 'balloon' help system to the initial design. However, functional list creation was not rejected as a lifecycle component as the existence of a application model could not be relied on for the development of future applications.

The existence of a design model, in the form of a the Text application, reduced the reliance of the design process on the function list. However, this was not considered to completely remove the value of this activity. Function list construction represented the specific consideration of the proposed module’s requirements. The aim of this test application had been limited to providing a demonstration of the development tool’s text handling abilities. Unlike the Standards module, it was not designed to compete with the paper alternative. Construction of the function list forced the consideration of the elements necessary to access the advantages of the medium as well as restating standard elements to ensure design consistency throughout the DAVE system. It is likely that these features would still have been considered if the project had moved directly to the design of a non-functional prototype. However, the function list activity was retained as a 'modular system' lifecycle activity as it provided an additional point of review, helping to formalise an otherwise loosely managed process.

7.8.3 Prototyping

Module design and construction followed the sequence set out in the prototype development cycle, (figure 5.8). The successful development and delivery of the Anthropometry module, based on this cycle, demonstrated an effective match between this cycle and the in-house scenario.

Figure 7.8 presents an amended model of the lifecycle's prototyping phase. The basic structure of the original prototype development cycle remained the same. A greater distinction was made between iteration at a prototype level and iteration at an element level. Added to the prototype level evaluation was a list of applied approaches to
evaluation. The evolving design of the prototype was continually evaluated through developer assessment and user feedback. This led to changes in the specification of the prototype's overall form and function. The focus of this level of design was the interface. Graphical elements could be quickly created, positioned and scaled within the prototype window. The non-functional graphical elements could then be selected for further development. The detailed design and the development of required code was then completed within the developer centred iterative cycles, at the element level.

Completion of an element occasionally revealed design issues which influenced the continued development of the module. For example, implementation of the retrieval code allowed assessment of the interface's text manipulation functions, which in turn highlighted the inadequacy of the context window width when accessing large documents.

The prototype development cycle did not attempt to provide design solutions. Instead, it provides a framework for design evolution. It extended the original prototype development cycle model, (figure 5.8), by further defining levels of design and highlighting the point at which developer, expert and user input provided effective support for design.

Whilst informal user feedback was incorporated throughout the prototype's evolution, expert evaluation was applied as a formal event. Expert evaluation for the development of the first module was criticised for its minor impact on the design of the product, (section 6.7.3.1). This was attributed to the late stage of evaluation at which point prototype evolution had revealed and corrected the major faults. Whilst it provided a safeguard against the delivery of an unsuitable product within the acceptance phase, it provided little contribution to the design of the module. Earlier involvement of the expert for the second module provided recommendations which were more significant and less specific. This suited the development approach by providing design goals without constraining the search for a solution. The relatively early stage of evaluation allowed criticisms such as overall complexity and interface standardisation to be addressed in the continuing evolution of the prototype.
A concern raised through consideration of the first application of the prototype development cycle was the selection of an appropriate point of transfer to the formal acceptance phase, (section 6.4.4). The early application of the expert evaluation process reduced the acceptance phase to the processes of user testing and possible design iteration. This placed user testing as the sole safeguard against implementation of an unsuitable design. The acceptance phase was initiated as soon as the module was able to demonstrate full functionality, but before it was considered to be deliverable by the developer. This approach followed the stated principle of accelerated transfer.

Whilst a prototype 'completeness' measure to indicate the most suitable point of phase transfer could be useful project management tool, the diversity of potential applications prevented the design of an adequate measure. Instead, the point of transfer relied on the judgement of the developer who was in the most informed position to assess design completeness.
7.8.4 Acceptance

The approach of direct observation for the second module's user test differed from the background logging approach used for the first module. Whilst providing effective usability measures, background logging was not considered to be an efficient measurement approach for user testing, (section 6.7.3.2). A similar conclusion was reached by Baber and Stanton (1996) who estimated the time required for video analysis to be longer than the recording period by a multiple of ten. The change to direct observation sacrificed the level of realism in the test environment achieved through background logging. In its place, the advantages of reduced development effort and the opportunity to assess user behaviour were gained. Whilst the impact of the presence of the developer in terms of user reaction to the system could not be measured, direct interaction was discouraged and the subjects' awareness of the observer during the task appeared to be low.

The key difference in the results achieved through user observation was the gathering of a list of observations in addition to user stated problems. This observation list revealed subtle interaction issues not recognised by the users. Observer presence was further capitalised on by the immediate interview of subjects following task completion. This supported the discussion of user stated problems and allowed the involvement of the individual in the analysis of their observed approaches to interaction. In this way, the follow-up interview acted as both a means of clarifying problems experienced by the users and instructing users in more efficient search strategies.

The group's initial specification of the prototype did not anticipate a future conflict between the paper and computer sources. By comparing the evolving computer tool directly against the existing paper source the evaluation recognised the competition inherent in the future delivery of the module. Performance using paper provided both a baseline for performance and also a demonstration of paper's advantages and limitations. By not pre-training the users, the evaluation attempted to mimic the anticipated occasional use of the tool. This approach tried to force user errors rather than test the prototype as a form of a pre-delivery 'rubber stamp' of acceptance. This takes the view that pre-delivery iteration is less costly than repair of the delivered application. The higher cost of repair may be counted in terms of the number of applications which comprise the newly distributed system and also in user confidence in the tool.
The results showed a close similarity in performance between paper and computer despite the 'worst case' evaluation of the prototype. Design enhancements recommended by the evaluation tended towards operational efficiency and interface intuitiveness. The key issue of search errors proved to be marginally better for the computer tool. These results offered the possibility of the tool providing an enhanced search capability following the implementation of design recommendations raised by the formal user test. The most promising feature of the new tool was that search performance should not be related to familiarity with the document as observed for performance using the paper source. This prompted consideration of widening the indexed text database to other areas, including the group's questionnaire and miscellaneous documents.

7.9 Conclusions

Development of the Standards module provided further evidence for the suitability of the evolutionary prototyping approach for tailored development of a product from a loose product concept. The results of the formal user evaluation confirmed the product's achievement of acceptable levels of functionality and usability. The delivered Standards module reached and extended its specification. The success of the product is considered further in chapter 10, based on its use over the full DAVE development project.

The conclusions drawn at this point in the DAVE development project are presented below.

1. Despite a growing confidence in user participation within the specification process, users remained unsure when asked to define functional details. The developer was best positioned to undertake the creative design process leading to an initial prototype.
2. The user group was able to actively respond to design ideas when presented in the form of a prototype. User centred prototype evolution produced an effective tool which was acceptable to the user population.
3. The existence of an application on which to model the prototype replaced the role of the function list.
4. Where boundaries to the data source are known, data set selection should be the first 'module level' specification activity. The data set may be refined further, based on assessment of the prototype's design.
5. Expert evaluation of an early semi-functional prototype provided a greater contribution to the prototype evolution process than pre-delivery expert evaluation. Early feedback allowed the design to address important 'general' design concepts.

6. Direct observation during user test completion required less analysis effort and gathered more detailed information than background event logging.

7. Development and delivery using a high level prototyping environment allowed major design changes to be made throughout the lifecycle.

8. User testing captured all critical design issues despite an accelerated transfer of the prototype to the acceptance phase.

9. The risk of prototype over-development remained during the prototype development cycle despite the stated intention of accelerated transfer to the acceptance phase.

The apparent success of the amended lifecycle in achieving a novel product which was acceptable to the users provided confidence in applying the model to further development. The main concern was the lack of firm structure with which the project could be monitored or controlled. The central recommendation for the continued development of the DAVE system was to:

- Consider the use of formal techniques to further structure the 'modular system' lifecycle.

Chapter 8 discusses the specification and development of the DAVE system's third module, the Vision module. Development of the third module provided an opportunity to compare and contrast the project's user centred prototyping approach with alternative development guidelines. The approach taken was the parallel development of two early prototypes following the prototype development cycle and the heuristic guidance of the HUFIT toolset respectively.
8 The Vision Module

8.1 Chapter Overview

Chapter eight discusses the design and development of the DAVE system's third module, Vision. Two prototypes were developed in parallel following separate design approaches. The design of prototype A followed the DAVE project's lifecycle of prototype evolution based on a minimal specification. In contrast, prototype B was designed against a specification determined following the HUFIT toolset (HUSAT 1989).

A revised lifecycle is presented incorporating the twin prototype approach to design. The specification and design processes of the parallel development phases are reviewed. Consideration of the design evolution of prototype A is followed by a review of the HUFIT guided specification and design of prototype B. The results of a paired evaluation of the two prototypes are presented and discussed. The preferred components were combined in a third prototype prior to product acceptance and delivery.

The project returned to the twin tasks of expert evaluation and formal user testing within the lifecycle's acceptance phase, as applied to the design of the first DAVE module. The increase from one to three individuals for expert evaluation is discussed, including the role of a heuristic list to guide the process. The results of formal user evaluation are presented and the module's levels of use immediately following delivery are considered.

The chapter's discussion of methods compares the separate prototype design approaches against the characteristics of the in-house development environment. The justification for the development of the Vision module despite the anticipated low levels of use is considered. Parallel prototype development followed by design integration is discussed as a lifecycle approach for further development projects. However, this is rejected as a viable approach within the given scenario. The approaches to expert evaluation and formal user testing are discussed and recommendations made for acceptance phase processes. The implications of the Vision module's development on the DAVE project are considered in a review of the 'modular system' lifecycle.

182
8.2 Lifecycle Approach

The separate evaluations of the DAVE system's first two modules provided the background to the development of the third module. They suggested that both products achieved acceptable levels of usability and user satisfaction prior to delivery. This was supported by observation and informal feedback gained through the in-house environment. These positive indications of success for the existing DAVE system's modules established the 'modular system' lifecycle as a suitable guide for this level of development.

Continuation of the DAVE development programme provided an opportunity to further tailor the design of the lifecycle model to the characteristics of the development environment, in conjunction with continued module development. Additional investigation of the lifecycle design needed to coexist with the overriding need to ensure the success of the development project. The lessons of previous module development, in the form of the evolved lifecycle methodology, provided the best available guidance for continued design. Consequently, this model was retained as the framework for the development of the next system module.

8.2.1 Parallel Prototype Development

The main development effort for the previous modules was placed in the prototype development cycle of the development phase. Development of the third module offered an opportunity to consider this phase in more detail. User acceptance of the delivered products confirmed the ability of the prototype development cycle to guide user centred design. The development of the third module of the DAVE system provided an opportunity to compare the prototype development cycle against an alternative approach to design. This was made possible by the collaboration between the developer and another member to the group to offset the added development effort.

The lifecycle was modified to include parallel specification and design processes to guide the design of two separate prototypes, (figure 8.1). The design features of the resulting prototypes could then be compared and combined to produce a final module for implementation within the DAVE system structure.

183
The design of Prototype A followed the 'modular system' lifecycle approach of minimal specification followed by prototype evolution. This process was carried out by the developer of the DAVE system. Prototype B was specified and designed following the guidance of the HUFIT (Human Factors in Information Technology) toolset. This process was carried out by a member of the user group with the assistance of the DAVE system's developer.
8.2.2 The HUFIT Toolset

The HUFIT toolset (Galer and Taylor 1989) consists of a set of procedures guided by the completion of proformas which provide a structured approach to the involvement of the user group in the specification process. The tools and their sequence of application are illustrated in figure 8.2.

Figure 8.2 The HUFIT Planning, Analysis and Specification Toolset (HUSAT 1989)
Table 8.1 presents the list of the HUFIT tools and provides a brief review of each tool’s function.

<table>
<thead>
<tr>
<th>HUFIT Tool</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Mapping</td>
<td>Record high level information about the target users of the proposed product.</td>
</tr>
<tr>
<td>User Characteristics</td>
<td>Record the characteristics and requirements of the target users of the proposed product.</td>
</tr>
<tr>
<td>Task Characteristics</td>
<td>Highlight a number of aspects of user tasks and their implications for the design of products.</td>
</tr>
<tr>
<td>User Requirements Summary</td>
<td>Summarise the results of the previous Requirements tool to feed into the design process.</td>
</tr>
<tr>
<td>Usability Specification for Evaluation</td>
<td>Setting product specific usability goals and their tests including their criteria, leading to a full evaluation plan.</td>
</tr>
<tr>
<td>Functionality Matrix</td>
<td>Cross referencing of user and task requirements with individual functionality items to assess trade-off decisions during product planning.</td>
</tr>
<tr>
<td>User Computer Interface Design</td>
<td>Interface design based on an overview of principles for good design.</td>
</tr>
</tbody>
</table>

Table 8.1 Components of the HUFIT toolset

8.3 System Specification

The concept of a software tool to support the group’s task of vehicle vision arc measurement grew from a group member’s design of a new data gathering technique (Judd 1995). The new technique used trigonometry in place of the group’s traditional method in which compass bearings were used to determine the vehicle’s vision arc. The change in technique was the response to a previous vision evaluation which had been disrupted by a magnetic disturbance.

Vision arc measurements were conducted on large areas of flat ground. The new technique extended reference ‘baselines’, of a fixed length, along each side of the vehicle. By means of radio communication, an observer within the vehicle directed the
positioning of markers on the ground at the extreme points of each crewstation’s visible arcs. The distance from the markers to the end of the nominated ‘baseline’ were then measured and recorded. The data was taken back to the group’s offices to be plotted on graph paper.

The development of the vision arc measurement technique into a computer based tool was proposed as the continued work programme for the technique’s designer with the assistance of the DAVE system developer. The group member’s restricted time-scale forced the overlap of the developer’s effort between this project and agreed DAVE work programme of upgrading the Anthropometry module, (section 9.4). This represented a deviation from the ‘modular system’ lifecycle in which developer effort had been limited to a single software project at any point in time. However, the circumstance of shared development effort for the proposed application was considered to justify this procedural change. This additional development effort provided the associated study with an opportunity to compare the ‘modular system’ lifecycle’s approach with an alternative method.

The HUFIT toolset was chosen as the most appropriate alternative approach to formally guiding product specification and design. The HUFIT toolset formalised the DAVE system’s development philosophy of involving all stakeholders in the early design stages. It is aimed at providing product design teams with tools to support the gathering of user and task information for the planning, analysis and specification activities leading to product design (Catterall 1991). Other software development approaches such as Ethics (Mumford 1991) and the Jackson Method (Sutcliffe and Wang 1991) were rejected as they were considered to require a level of resources beyond the scope of this project.

Whilst the HUFIT toolset endorsed subsequent design iteration, it placed a focus on careful specification as a necessary starting point to design. This contrasts with the approach to the DAVE project’s previous development projects which placed a greater emphasis on user centred design iteration as a means of overcoming the limitations of minimal specification. The aim was to compare the product of detailed specification, (prototype B), against a product rapidly evolved from a loose concept, (prototype A), as a means of assessing the relative benefits of the alternative approaches to initial design.
8.4 Prototype A - Modular Development Lifecycle

8.4.1 Module Specification

Prototype A’s specification phase relied on the same activities described for the previous modules. These activities included data set selection, requirements definition and function list construction. Whilst the first two DAVE system modules can be classified as information retrieval tools, the Vision module’s primary role was considered to be data gathering. However, this expanded to include office based retrieval of vision arc plots from a database of past vehicle assessments.

‘Module level’ specification of the preceding modules had been guided by data set selection. The form and extent of the data provided limits within which module functions could be proposed. Whilst data set selection was retained, it was the data to be gathered rather than the data to be retrieved which was determined for the Vision module. As an existing data set was not available as a starting point for design, specification of the vision module needed to start by determining the module’s functional aims. The design could then ensure that all required information was gathered.

The group’s requirements for the proposed tool were determined through the developer's discussion with the individual group members. The resulting requirements definitions are listed below:

- Guide the trigonometric method of vision plot data gathering.
- Graphically display the gathered data, in the field.
- Flag out-of-range measurement values.
- Construct and output a vision arc plot of the gathered data.

As each function was linked to a specific data requirement these functions enabled the developer to determine a database structure to serve the application. This structure was used in the final product.

Function list creation was considered to be effectively replaced by the existence of a design model on which to base an initial prototype, (section 7.8.2). As a design model was not available for the Vision module, the developer centred creation of a function list...
was retained as one of the 'module level' specification activities. The resulting list is presented in table 8.2.

<table>
<thead>
<tr>
<th>Application Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Input vehicle details</td>
</tr>
<tr>
<td>2 Define baseline limits</td>
</tr>
<tr>
<td>3 Define workstation positions</td>
</tr>
<tr>
<td>4 Select a crewstation</td>
</tr>
<tr>
<td>5 Select a vision device</td>
</tr>
<tr>
<td>6 Select type of head movement</td>
</tr>
<tr>
<td>7 Select the main armourment position</td>
</tr>
<tr>
<td>8 Input measurements</td>
</tr>
<tr>
<td>9 Mark alternative vision arcs</td>
</tr>
<tr>
<td>10 Display vision arc plot</td>
</tr>
<tr>
<td>11 'Out-of-range' warning</td>
</tr>
<tr>
<td>12 Plot measures to screen</td>
</tr>
<tr>
<td>13 Module title</td>
</tr>
<tr>
<td>14 Return to entry screen</td>
</tr>
<tr>
<td>15 Log events</td>
</tr>
<tr>
<td>16 Display time/date</td>
</tr>
<tr>
<td>17 Quit from module</td>
</tr>
<tr>
<td>18 Help (on-line)</td>
</tr>
</tbody>
</table>

Table 8.2 Vision Module Function List

8.4.2 Prototype Evolution

A basic prototype design was created by the developer based on the minimal specification offered by the lifecycle's 'module level' specification phase. This design was evolved following the lifecycle's prototype development cycle through informal user feedback.

The design conformed to the functional specification of 'field based' data gathering by restricting the interface size to the area available on the group's recently acquired portable computer. Incorporation of the specification's listed functions led to the spread of functions between three screens. The design passed through multiple variations.
Figures 8.3 and 8.4 represent a snap-shot from the sequence of prototype evolution at the point of user testing.

The design approach for the information screen, (figure 8.3), was to guide the user through three input sections. The data gathered through these sections was automatically translated into a plan diagram which illustrated the new technique’s measurement grid and provided visual feedback for the workstation locations relative to the vehicle.

![Vehicle Information](image)

**Figure 8.3 Information Input Screen (Prototype A)**

Prototype A’s data input screen, (figure 8.4), split the functionality into four sections. The top button bar remained constant between interface screens, whilst highlighting the identity of the present screen. The subject was required to choose the workstation, vision device, head movement and gun elevation for each vision arc. The application responded to this information by illustrating the vehicle and the workstation’s position within the measurement grid. The user could then either enter the values directly or click on the approximate position of the arc point to be given an estimate of the expected values. The diagram was redrawn whenever values were entered to provide immediate visual feedback.
Figure 8.4 Data Input Screen (Prototype A)

The function of data gathering and plot construction fulfilled the functional specification for the module. The addition of a button to move to a separate screen for database retrieval and display went beyond the specification. It was included as a prototype option by the designer. Addition of this button involved little additional design effort. However, in doing so it raised user feedback concerning the requirement for this potentially useful feature.

8.5 Prototype B - HUFIT Toolset

8.5.1 Module Specification

The design of prototype B was guided by the Planning, Analysis and Specification (PAS) tools of the HUFIT toolset. These tools provided a similar approach to the specification of prototype A, formalising the gathering of user specific information for which the primary source was the specified user population.

The tools provided a consistent structure and format for information gathering in support of the process of planning, analysis and specification. The tools guided user centred information gathering under the headings of background details, user group
characteristics, use characteristics, task environment, task characteristics and concluding in a user requirements summary, (appendix D).

The information gathered through the HUFIT tools was applied to the toolset's functionality matrix, (appendix E). The matrix was used as the guide to the design of prototype B. The task and user characteristics were entered in the grid structure for direct evaluation against the characteristics of the proposed design's functionality, software components and hardware platform.

8.5.2 Prototype Design

Both prototypes relied on information from the same user group and needed to conform to the same hardware and data boundaries. This led to inevitable similarities between the two prototypes. As the design processes ran in parallel it is difficult to separate the source of common design features. One example is provided by the common number of screens between prototypes. This may be due to the transfer of ideas or simply be the most obvious solution to the hardware restriction of the portable computer's screen size. At the same time, many functional aspects of the two prototypes show separate creative approaches.

The design approach taken for the information input screen of prototype B, (figure 8.5), split the task into two sections. The left of the screen required the user to input background data and make a selection from the pre-defined workstation categories. The neighbouring section incorporated a non-functional, stylised diagram of a vehicle indicating the baselines. Two tables, positioned below the diagram were included to gather the end point co-ordinates of the baselines.

Prototype B's data gathering interface, (figure 8.6), took a 'data card' approach to the input of measures. On the left of the screen, the user could choose from scrollable lists of pre-defined workstations and vision devices, head movement and gun elevation. The user indicated the relevant baseline on a stylised diagram. These actions would select a blank data card on which the user could type the co-ordinate measurements plus any comments. Included below the data card window was a numerical input pad including the abbreviations for the possible types of arc point.
Figure 8.5 Information Input Screen (Prototype B)

Figure 8.6 Data Input Screen (Prototype B)
8.6 Comparative Evaluation

8.6.1 Aim

From the development project perspective, the object of parallel prototype development was to determine a suitable design for the final product. Both prototypes achieved the same final goal of measurement gathering through different interface approaches. The task of prototype comparison was not undertaken to choose which prototype to develop further but rather to choose which features of the two prototypes to include in a third, combined prototype. A formal user test was devised to test the usability of the central functions of the two prototypes. Acceptance criteria were not set at this point as the evaluation was aimed at design comparison, as opposed to final product acceptance.

8.6.2 Method

The approach taken to evaluation was comparative user testing of the two prototypes followed by a semi-structured user interview. The evaluation used all of the eight designated members of the specified user group. Half of the subjects were randomly assigned to using prototype A followed by the use of prototype B. This order was reversed for the remaining subjects.

The prototypes were presented on the group's portable computer within an office environment. A task sheet was used to deliver the test instructions. Each subject was asked to conduct a vision assessment for two vision devices. The task required the user to set up the device details on the data input screen of the prototype and request data-points from the experimenter via a hand-held radio. Completion of the task using one prototype was followed immediately by completion of an equivalent task using the second prototype. In each case the user was provided with one incorrect measurement value which they were asked to correct at the end of the test.

Following test completion, both prototypes were placed before the subject. The subject was then taken through a questionnaire, adapted from Ravden and Johnson (1989), (appendix F). The subject was encouraged to explore the prototypes further when answering the questions and to discuss issues with the observer.
8.6.3 Results

The results from the vehicle evaluation fell into two categories. The first was the information gathered through subjective user response. Objective data was gathered in the form of total completion time and errors.

Questionnaire results showed a preference for prototype A in the areas of feedback, data input and aesthetic appeal. Conversely, the users preferred prototype B in the areas of consistency, compatibility and structure. User opinion in the areas of simplicity and functionality was split between the two prototypes. The final question of an overall preference revealed group support for prototype A. The spread of subjective feedback confirmed the need to combine preferred features. However, most of the preferred features were drawn from prototype A.

![Chart showing task completion time and standard deviation](image)

**Figure 8.7 Evaluated Prototype's Task Completion Time and Standard Deviation**

The completion times, (figure 8.7), showed a relatively short time for entering background information and accessing the database screen for prototype A and a longer time for data input. Statistical analysis using the related t-test (n=8) showed only the difference in background information input to be significant (p<0.05). This data must be treated with caution due to the nature of the task and the low number of subjects. It has been included to support the subjective responses of the subjects leading to a decision concerning the direction of future design evolution. The objective results
supported prototype A’s guiding approach to data input. Whilst the users expressed a preference for the prototype A’s graphical approach user problems were reported related to processing hardware’s ability to meet its processing requirements. It was therefore necessary to redesign of this feature to overcome the performance limitations.

The second source of objective data drawn from the evaluation process is presented in table 8.3. An observation log of actions was kept during the completion of the evaluation tasks. The table has been divided into the two sections of errors and requirements for help.

Subject errors were categorised as either mechanical or non-mechanical. Mechanical errors included any error immediately corrected by the subject, such as miss typing. Non-mechanical errors were considered to result from a user misunderstanding, such as trying to enter measurement data without first specifying the workstation details in the input screen. Whilst both prototypes produced a high level of errors, this was anticipated at this early stage of design development. The relative acceptability of Prototype A was further confirmed by the lower observed error level.

<table>
<thead>
<tr>
<th></th>
<th>Prototype A</th>
<th>Prototype B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-Mechanical Errors</td>
<td>30</td>
<td>44</td>
</tr>
<tr>
<td>Mechanical Errors</td>
<td>13</td>
<td>24</td>
</tr>
<tr>
<td>On-line Help Usage</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>Requests for Assistance</td>
<td>5</td>
<td>7</td>
</tr>
</tbody>
</table>

Table 8.3 Total Usage Errors and Help Enquiries for Evaluated Prototypes

The number of help enquiries was taken as a second measure. These events included access of each prototype’s ‘on-line’ help system and direct requests to the experimenter for assistance. This data provided an indication of user confidence when using the prototype. The results showed a greater requirement for assistance using prototype B and a reluctance to use the on-line help for prototype A despite both help systems adopting the same ‘balloon’ help mechanism.
8.7 Design Integration

The comparative evaluation of the two prototypes provided the designers with a user centred evaluation of the prototype components. This provided the basis for the selection of preferred components for the next stage of the project, development of a combined prototype. The framework for prototype A provided the structure for the continued development of the prototype. Those components of prototype B preferred by the users were then incorporated into the design to produce a prototype C.

Following the integration of the chosen prototype components within the new interface screens, the DAVE system style and format was applied to each interface screen. A detailed background graphic incorporating each screen’s full functionality was used. This followed the successful application of this approach for the preceding modules.

Figure 8.8 shows the revised information input screen. The vehicle information fields were located in a separate ‘raised’ section. Prototype A required the user to confirm data entry, using an “OK” button. In its place, the direct entry approach of prototype B was adopted with the addition of automatic ‘tabbing’ through the fields in a recommended order. Workstation selection was simplified by reducing user choice to a standard set of crewstations. Completion of the vehicle dimensions and crewstation location coordinates resulted in the automatic drawing of the baseline scale and included the graphical indication of workstation position, relative to the grid.

The design of the data input screen, (figure 8.9), followed the user preference for the graphical feedback whilst recognising the performance limitation shown through direct graphical interaction. The design compromised with the concentration on direct input of measures plus the immediate feedback of arc drawing in a separate window. Other sections of prototype A’s interface were retained with the addition of prototype B’s alternative numerical keypad. User confusion surrounding the data card approach of prototype B led to the use of a single data window.
A design for a Plot Output screen, (figure 8.10), was required to meet the needs of database retrieval and presentation. The code and interface components for plot creation were copied from the data input screen. The approach taken was the retrieval and plotting of the data, as opposed to the simple storage of completed plots. This allowed the
added functionality of selective arc display. The function of data category sorting was added after a requirement was shown during the acceptance phase.

Figure 8.10 Vision Module Plot Output Screen

8.8 Acceptance

The previous sections have described a departure from the project's normal lifecycle approach. This allowed a comparison of methods for the development of DAVE system modules to be made. Completion of the combined design in the form of prototype C marked the end of this deviation with a return to the 'modular system' lifecycle's formal acceptance phase.

As before, the role of the acceptance phase was intended to confirm the design's overall fitness for purpose and to raise possible improvements. Evaluation of the Anthropometry and Standards modules had been based on both expert evaluation and formal user testing. Evaluation of prototype C incorporated both of these activities.
8.8.1 Expert Evaluation

The minor contribution provided by expert evaluation to the development of the first module was attributed to over-development prior to evaluation, (section 6.7.3.1). Expert evaluation had a greater contribution for the second development project as part of the design process, (section 7.8.3). The exclusion of this technique from the prototype development cycle of prototype A was made to ensure commonality between the information sources applied to the parallel design techniques. However, it was still considered to be a beneficial activity. Consequently it was included as an acceptance phase activity to evaluate prototype C.

The approach to this technique was altered to overcome the limitations of the technique observed following expert evaluation of the Anthropometry module prior to delivery. The number of experts were increased from one to three, each with experience in the development of software for the organisation. This increase in numbers was not made as a formal recommendation at the end of the Standards module development cycle as the availability of additional experts was not anticipated. Another change to the previously adopted approach was the insistence that the expert panel follow a pre-defined list of design heuristics in order to co-ordinate the process. The heuristic list chosen, (table 8.4), was designed by Molich and Nielsen (1990). The design recommendations elicited by this approach to expert evaluation provided applicable guidance for the continued design of the prototype interface. All recommendations made at this point were applied to the prototype prior to formal user evaluation.
1. **Heuristic**

   **Expert Recommendation**

1. Simple and natural language - Simplify button labels
2. Speak the user's language - OK
3. Minimise user memory load - Rearrange workstation input fields
   - Add interaction to the data input illustrative graphics.
4. Be consistent - Ensure left to right, top to bottom completion sequence
   - Mirror baseline selection buttons with baseline orientation
   - Ensure labelling consistency
5. Provide feedback - Mark pressed buttons with inverse video
   - Prompt next button group in sequence
   - Fill plotted arc areas
6. Provide clearly marked exits - OK
7. Provide shortcuts - OK
8. Prevent errors - Prevent entry of inappropriate vehicle dimensions
9. Good error messages - OK

<table>
<thead>
<tr>
<th><strong>Table 8.4 Expert Evaluation Recommendations (Molich and Nielsen heuristics)</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Heuristic</strong></td>
</tr>
<tr>
<td>1. Simple and natural language</td>
</tr>
<tr>
<td>2. Speak the user's language</td>
</tr>
</tbody>
</table>
| 3. Minimise user memory load | - Rearrange workstation input fields
   - Add interaction to the data input illustrative graphics. |
| 4. Be consistent | - Ensure left to right, top to bottom completion sequence
   - Mirror baseline selection buttons with baseline orientation
   - Ensure labelling consistency |
| 5. Provide feedback | - Mark pressed buttons with inverse video
   - Prompt next button group in sequence
   - Fill plotted arc areas |
| 6. Provide clearly marked exits | - OK |
| 7. Provide shortcuts | - OK |
| 8. Prevent errors | - Prevent entry of inappropriate vehicle dimensions |
| 9. Good error messages | - OK |

**8.8.2 Formal User Evaluation**

Formal user evaluation of prototype C was undertaken by measuring the vision arcs of a real vehicle using the fully functional module. All eight members of the user group completed the task. The evaluation followed the same format applied to the comparative evaluation of prototypes A and B. Task completion time was disregarded as this data was considered to hold little value for the evaluation of a single design for which the acceptance criteria centred on task accuracy.

The revised prototype was demonstrated to each subject before moving outside to the vehicle. The subject was seated in the commander's workstation and asked to measure the arcs of vision for two of the vehicle's episcopic vision devices. The movements of two assistants outside of the vehicle were controlled by the subject using a hand-held radio. An experimenter was present within the vehicle to record the subject's operational...
procedure. When the subject had completed the vision assessment they were asked to close down the application and return to the office where a questionnaire was administered. The format was based on a usability questionnaire devised by Shneiderman (1992) with the addition of a section which sought further comments from the users, (appendix G). The questionnaire’s nine point scale was used to derive a ‘usability score’. Features scoring below five on the scale were classed as essential improvements prior to delivery. Features raised by the questionnaire included the need for a printing facility and the hardware issues of screen contrast and operating speed. The print facility was added prior to implementation. The problem of screen contrast and operating speed were subsequently addressed by upgrading the computer platform.

<table>
<thead>
<tr>
<th></th>
<th>Prototype C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-Mechanical Errors</td>
<td>18</td>
</tr>
<tr>
<td>Mechanical Errors</td>
<td>6</td>
</tr>
<tr>
<td>On-line Help Usage</td>
<td>0</td>
</tr>
<tr>
<td>Requests for Assistance</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 8.5 Total Usage Errors and Help Enquiries for Prototype C Evaluation

Table 8.5 presents the objective performance measures of errors and help enquiries. The operation errors, as compared against errors for prototypes A and B, were greatly reduced. More importantly, the suitability of the prototype design was confirmed by the error rate in the final data set gathered by each subject. Despite some errors in the data gathering process, each error was either corrected by the subject or had no direct effect on data integrity resulting in a 100% accuracy level for each subject’s completed data set. A further indication of the acceptability of the design was the users’ absence of need to access the on-line help system. The two enquiries made to the experimenter were related to the vision evaluation task as opposed to the direct operation of the prototype.
8.9 Delivery and Maintenance

The completed module was designed to be a mobile data gathering tool. This reduced the task of product delivery to the installation of the iterated module on a designated portable computer. In addition to this basic task, the module was implemented across the group's network making it accessible from the desktop computer of each group member. Initially, the purpose of this distribution was to provide potential users with the opportunity to refresh their memory concerning the measurement technique and in the procedural requirements of the software. In the longer term, the distributed module was provided as a networked resource to allow users to retrieve and compare vision data previously gathered using the portable tool.

Following delivery there was no direct requirement to undertake a vision assessment as part of the regular work programme. Despite this absence of an immediate work programme requirement, the lifecycle's post-delivery iteration process was called upon by users exploring the application on their desktop computers. This led to the prompt correction of a minor graphical problem within the display screen.

The absence of a direct work programme requirement for the vision tool continued following delivery with a change in direction of the group's work programme related to a change in staff numbers. The change in group composition, in conjunction with a rise in work programme requirements in other areas, was sufficient to prevent the group from undertaking further vehicle evaluations. Whilst the requirement for the tool was considered to be periodic when assessed at the 'system level' specification phase, the shift in the group's work programme was not foreseen. The very specific nature of the tool made it vulnerable to both the market requirement for vision assessment and the organisational environment which determined the effort available for vision assessment, in competition with other work areas.
8.10 Discussion of Methods

The Vision module’s ‘parallel prototype’ lifecycle model, (figure 8.1), represented an alternative to the lifecycle forms previously applied to the DAVE project. The lifecycle’s design was made possible by the addition of an individual to undertake parallel development. The main value of parallel design for the DAVE development project was considered to be the opportunity it provided to advance the project’s approach to development through direct comparison with an alternative.

Figure 8.1 illustrates the difference in emphasis between the parallel approaches to product specification and design. The design of prototype A involved the brief determination of a minimal specification for the proposed product which was then refined through the ‘modular system’ lifecycle’s iterative development cycle. In contrast, the design of prototype B took the HUFIT toolset’s structured approach to specification and design in an attempt to ensure the acceptability of the initial design. The project was then able combine the best features of the two prototypes in a final product.

Both design approaches produced a prototype which met the central goal of guiding the new vision arc measurement technique. However, formal user evaluation of the two prototypes revealed an overall preference for prototype A. In addition, the objective results showed higher error levels for prototype B and a greater requirement for assistance. It was concluded that the design of prototype A matched user requirements more closely than prototype B. This led to prototype A’s selection as the basis for the final product with the addition of any preferred features from prototype B.

Before these conclusions can be extended to include the alternative approaches taken to specification and design, it is necessary to consider any inequalities between the parallel design processes. Both were completed within the same time-frame, using the nominated user group as the source of design guidance. The main difference lies in the characteristics of the two developers. Both individuals had a similar formal training background. Whilst individual difference was unavoidable, it must be taken into account in considering the paired prototypes. The developer of Prototype A had the added experience of developing the existing modules of the DAVE system for the group. The developer of Prototype B had less development experience and relied on the HUFIT tools.
for guidance. Conversely, the designer of prototype B had an intimate understanding of the new vision arc measurement technique. Although the form and features of prototype B were under the full control of its designer, it was necessary for the developer of the DAVE system to provide assistance for its construction. It was not possible to completely exclude influences between the parallel processes due to constraints of the project environment. However, this risk was recognised and measures were taken to minimise this effect. These included the use of independent offices and separate user feedback meetings. Whilst an unrestricted transfer of ideas between the two designers might have assisted the design process, it would have reduced the need for both designers to explore creative solutions. By minimising this transfer the project demanded that both designers independently tackle similar problems. The project was then able to choose between the alternative solutions through formal evaluation. In addition, this approach allowed the parallel study to consider the competing design methods.

8.10.1 Programme Selection

Selection of the Vision module as a development project arose as the continuation of a group member’s project to develop the group’s vision arc measurement techniques. This ‘short circuited’ the ‘modular system’ lifecycle by selecting and initiating the development of a computer based tool before completion of the main DAVE system development cycle, to update the Anthropometry module, (section 9.4). Although this conflict was recognised, initiation of the project was a response to the availability of another individual to meet part of the associated development effort. This situation arose outside the control of the DAVE system’s developer, illustrating the need to respond to organisational demands which do not fit within the predicted structure of a defined lifecycle.

A further contrast with the system level specification phases of the first two modules was the endorsement of a concept product without the group’s formal consideration of alternative development projects. Development of the new vision arc measurement technique offered a natural progression for the group member’s work programme. Whilst the risk represented by the periodic nature of the work programme’s demand for vision assessment was recognised, development was justified by the perceived benefits of the tool and the absence of alternative projects for the second developer. Although the latter issue does not justify the selection of the development work programme, it
remained as an organisational reality. The advisability of selecting this development project is considered in more detail in chapter 10, in light of the cost of development and the results of product usage since delivery.

8.10.2 Prototype Specification

Having agreed a concept product based on the new vision arc measurement technique the specification and development phases were split into parallel processes, (figure 8.1). Prototype B’s designer applied the HUFIT toolset to specify the product as a guide to the design. In contrast, the DAVE system’s developer derived a minimal specification, relying on the prototype development cycle to guide the design of prototype A. Whilst the HUFIT specification process continued, an initial design for prototype A was produced. This prototype was then iterated based on the input from all group members. At the same time, prototype B’s design process concentrated on product specification. The HUFIT toolset’s ‘Background’ and ‘User Group Characteristics’ tools sought to generalise the characteristics of the group to provide an overall profile. This generalisation becomes necessary for larger development projects in which the full user group cannot be involved in the design process. In comparison, the design and iteration of prototype A was able to consult the full population within the iterative design process.

Following user description, the HUFIT toolset was used to characterise aspects of the proposed tool and its associated task. The HUFIT tools applied included the ‘Task Environment’, ‘Use Characteristics’ and ‘Task Characteristics’ tools. The ‘Task Environment’ tool was particularly relevant for the design of the Vision module. It ensured the consideration of the operational environment features such as data gathering in the field and its use in conjunction with a hand held radio. The ‘Use Characteristics’ tool confirmed the anticipated low level use of the tool, (appendix D). However, this prediction did not dissuade the group from assigning development resources to the project.

Application of these HUFIT tools was simplified by the existence of a defined technique for vision arc measurement on which the product could be based. Consideration of the paper based task assisted completion of the tool proformas. In comparison, task specific issues were raised during the discussions to determine the requirements definition for prototype A, but were not formally recorded. The short list of defined requirements
provided the only formal record of prototype A's specification process. From this point the prototype itself could be considered to be the specification 'document'. The evolving design of this prototype represented changes in the specification arising from consideration of the prototype's previous version.

The HUFIT approach to recording the requirements for the product's design culminated with their combination using the 'User Requirements Summary' tool. This directed a high level review of design requirements in the areas of hardware, software and training. These headings lead to the definition of standards which a design must reach to achieve acceptability. Whilst useful as a basis for evaluation of the product design, it did not provide direct support for the designer's immediate task of prototype creation. The designer of prototype B was left with the task of producing a design which did not conflict with any of the specified requirements. Design of prototype A approached from the other direction by producing a design which was iterated to overcome problems as they were revealed by user interaction. One advantage of this second approach was the existence of a design model upon which the group were able to base their comments. Associated with this was the risk of either failure to reveal a fundamental design problem or the failure of the iterative process to correct the problem without incurring an unsustainable development cost. At the same time, the approach of detailed specification did not guarantee the avoidance of these pitfalls.

Although the HUFIT tools set delivery standards, they provided the developer of prototype B with little direct design guidance. Prototype design at this small scale would have been supported more effectively if greater concentration had been given to the specification of functional goals within the 'User Requirements' tool. The tool was capable of representing the requirements in this form. The problem lay in its application. Consideration of the product's 'Hardware requirements', as directed by tool headings, were allowed to dominate the 'User Requirements Summary' in place of the less prescriptive 'Software design' headings of style, instructions and language/images. However, an improved balance would not have removed the risk that, in the absence of a model application, the users could experience difficulty in generating a common concept of the proposed product on which to base their specification.

The final HUFIT tool applied to the planning analysis and specification of prototype B was the toolset's 'Functionality Matrix', (appendix E). Its role was to cross-reference
user and task requirements against the proposed product functionality in determining the final functional specification before design. The value of this matrix in the design of prototype B was limited by the form of the user requirement summary. Completion of the matrix required the specification of the prototype’s basic functionality and interface design principles. These features had not been fully determined by this point due to the absence of a common concept between the prototype’s designer and the user group. Consequently, the matrix completed for prototype B included an incomplete mix of functional requirements and design style recommendations. The functional matrix became a complicated representation due to the large number of product elements and characteristics. Whilst the matrix offered a suitable method of considering the hardware set-up for the proposed product, affordable technology had already defined hardware solutions available to the project. It was considered that whilst the functionality matrix might support the evaluation of an existing product for implementation, it was not an appropriate tool for supporting the design process within the project’s development scenario.

8.10.3 Prototype Development

Both prototypes were constructed using SuperCard, thus ensuring an equal degree of development scope and flexibility for design manipulation. However, the more detailed approach to the specification adopted for prototype B led to the use of SuperCard primarily as a tool for creating and arranging the components suggested by the specification process. In contrast, SuperCard was stretched as an interactive tool for design presentation, user interaction and iterative alteration within the design process applied to prototype A.

Little reliance was placed on the specification of prototype A, depending instead on user centred evolution to clarify and develop the product specification in the form of a functional prototype. This approach relied on the flexibility of the prototyping tool, the availability of the users and their willingness to donate their time to providing design feedback. As reported for the development of the previous modules of the DAVE system, the interactive prototype provided a medium accessible to both the developer and the users. The prototype was a tangible representation of the current state of the product concept, as interpreted by the developer. Its accessibility helped to pool the groups ideas. At the same time, the flexibility of the tool did not constrain the design of the
prototype's initial form. Over this evolutionary period, features such as directed input, automatic baseline calculation and graphical feedback of entered data were incorporated and reviewed by group members. As these features extended the functionality beyond the prototype's specified function list it was possible to assess the impact of design changes on its usability. Some additions were able to reduce complexity. For example, automatic baseline calculation reduced interface complexity by removing the need for eight input fields. In comparison, the design of prototype B relied on the designer's interpretation of the HUFIT tools' text based representation of the user requirements list and functionality matrix to channel the design process.

A concern related to the direct comparison of prototypes A and B remains the individual differences of the two designers. The evolutionary approach to design required the designer to respond to user highlighted issues with creative design solutions. It is likely that another individual following prototype A's approach would have arrived at a different design. Similarly another individual might interpret the output of the HUFIT tools in a different way to the designer of prototype B. The central issue lies in the ability of the respective design processes to meet the user's requirements. By cycling until user acceptance was achieved, the prototype development cycle ensured an acceptable product. This influence of the individual was in the form of the solution and the effort required to reach it.

The initial design of prototype A provided the core functions of the concept product by making developer centred decisions concerning functional grouping, operation sequence, etc. This early design did not include the main features which made it more acceptable to the users than prototype B. It was the functional enhancements added through the evolution of the prototype which distinguished its design from prototype B. Consequently, the suitability of prototype A's design can be primarily attributed to the user centred iterative process, rather than the individual skills of the designer.

The influence of the individual can be seen in the design of prototype B. Few of its features can be related directly to the HUFIT specification process. Its guidance was general, recommending issues such as simplicity and interface consistency. The contribution of the HUFIT tools to specification was in ensuring consistency in user involvement when considering key issues. However, there remained a considerable gap between recognising important design issues and translating them into an actual product.
design. This relied on the skill of the designer. Prototype A's approach provided a means of confirming the proposed design solution in a constructive manner through user interaction and feedback.

By structuring the assessment of the users, task and environment, the HUFIT tools provided consistency in the specification process. However, the results of the parallel prototype development suggest that for this development scenario, the emphasis should be placed on user centred prototype evolution to arrive at an acceptable design. An important contribution offered by the HUFIT tools for larger development projects is the fostering of a closer working relationship between the designers and users when these groups are separated. As the in-house development of small applications automatically builds this relationship, the additional contribution of the HUFIT tools within this scenario is reduced.

8.10.4 Design Integration

Parallel prototype design has been reported as a suitable approach to design selection (Keyson and Parsons 1990). Up to this point, the development of the DAVE system relied on the development effort of a single developer. The development of two Vision module prototypes could not have been attempted without the availability of a second individual to undertake prototype development. Whilst the combined product incorporated aspects of both designs, the contribution of twin prototype design was minor in comparison with the added effort involved. Any benefit gained was due to the distinctly separate approaches of the two developers. Under the normal DAVE development scenario of a single developer this separation is unlikely to be achieved. The experience of the previous module design processes showed that design alternatives could be explored using a single evolving prototype. The development of several separate prototypes by an individual might be justified to objectively distinguish between fundamentally different application designs. However, this would considerably lengthen the design time-scale and must therefore be balanced against the importance of the proposed product and is therefore not recommended as an alternative to the 'modular system' lifecycle model.

Completion of a third, combined prototype brought the development process sequence back into line with the 'modular system' lifecycle approach. The informal involvement
of the user population within the prototype development cycle of prototype A continued to assist the design process providing both design feedback and encouraging user support for the product.

8.10.5 Expert Evaluation

The return of expert evaluation as an acceptance phase activity was the consequence of its omission from the design of prototype A, to ensure consistency between the parallel design processes. The value of expert evaluation for the Standards module led to its retention for the evaluation of the combined prototype prior to formal user testing within the acceptance phase.

In light of the limited contribution to the acceptance phase noted for the Anthropometry module, the approach to expert evaluation was changed to include a panel of three experts in place of the single expert used for previous evaluations. This presented the problem of control over the evaluation. When using a single expert for the evaluation, a self paced walk through was appropriate. Evaluation by a panel required some form of control over the process. The approach chosen was the use of the Molich and Nielsen (1990) list of heuristics. An alternative approach would have been the designation of a chairperson to guide the walk-through. The heuristic list was preferable as it both prompted consideration of important design areas and provided an agenda for discussion.

The revised approach to the expert evaluation raised a wide range of issues based on the heuristic list. In addition, the process was able to reject inappropriate design issues following informed discussion amongst the panel members. The comparative increase in effectiveness, may be due to the added complexity and accelerated development process of the Vision module. The downside of this approach to expert evaluation was the logistics of bringing the experts together for the evaluation and their cumulative cost. Whilst offering benefits over single expert evaluation, the choice of expert panel evaluation at this level of development must depend on the resources of the project.
8.10.6 Formal User Testing

The recommendations made by the expert panel were incorporated in a revised design prior to formal user testing. Formal user evaluation provided the final safeguard against the delivery of an inappropriate design. User testing of the combined prototype followed a similar approach to the evaluation of the Standards module. The advantage of direct observation was combined with the benefits of remote logging. The data files produced by the prototype during the evaluation task were compared against the correct measures to provide a performance measure.

The main difference between the user evaluation of prototype C and the user evaluations of the previous module prototypes was the procedure used to gain feedback. The approach taken for the previous modules was user/developer discussion through a semi-structured interview. These interviews were based on the specific functions of the tested prototype and observations made by the experimenter. They resulted in free form statements from the users concerning a wide range of design issues.

The questionnaire for the evaluation of the combined prototype combined the previously applied semi-structured questioning with a numerical score approach. The user completed usability scales aimed to quantify user opinion as a method for revealing and prioritising any design iterations required before delivery. The added section looked to prompt user discussion with the developer concerning improvements required before delivery. Whilst the numerical score approach may have been appropriate for gathering an impression of product usability for a large user population it had little direct application to the Vision module project. The small user size allowed full consideration of each user's views, as gathered by the additional questions. This led to final design changes to be implemented before delivery.

As reported for the previous module evaluations, direct observation and subjective user responses provided the richest source of recommendations for design iteration. Objective data in the form of accuracy, errors and help requests did not reveal specific design requirements. Instead they provided the primary criteria for product acceptance. The low number of operation errors demonstrated an improvement in operator efficiency. The 100% accuracy rate for the completed task showed these errors to be
non-critical. This supported the final conclusion that full module implementation could follow the iteration process in which user specified design requirements were included.

8.11 Lifecycle Overview

For this project's scenario of in-house development for a small user group, the approach of minimal specification followed by user centred prototype evolution was more effective than the more detailed specification provided by the HUFIT toolset. It is concluded that integration of the HUFIT tools within this study's modular system lifecycle would not offer a development advantage over the existing approach to specification. Consequently, the module specification and development phases of the lifecycle remain unaltered. However, it is considered that the HUFIT tools would provide particular benefits to development projects under any of the following conditions:

- Projects unable to adopt user centred prototype evolution for design.
- Large projects in which the development is completed by a team.
- Projects in which the user group is remote from the development individual/team.

As these conditions apply to a large proportion of regular software development, the role of the toolset should be assured.

The parallel prototype development approach to the initial design of the Vision module allowed the two alternative approaches to influence the eventual product design. This hybrid design benefited from the ability to choose between tested design features from each prototype. However, it was concluded that under the regular development resources available for the DAVE project, the added effort required by this approach could not be justified. Consequently, the approach of parallel prototype development has not been incorporated in the 'modular system' lifecycle.

The prototype development cycle was retained as the core process within the modular system lifecycle. Its application in the development of prototype A, and in the continued evolution of the combined prototype, was confirmed as an effective approach to design exploration. The processes of gathering requirement definitions and the construction of a function list provided an appropriate level of support for the task of initial prototype
design. It is noted that the success of this approach was linked to the flexibility of the software development tool and the continued willingness of the user group to provide informal feedback to the emerging design. User review of the prototype continued to concentrate on the broad design issues as opposed to the details of its design.

The return of expert evaluation to the acceptance phase of the lifecycle provided improved feedback, as compared to its first application as an acceptance process. This was attributed to the increased number of experts, the use of heuristics to guide the evaluation and the increased complexity of the prototype under evaluation. It was considered that the effective guidance provided by expert evaluation, in conjunction with the options of a panel and a heuristic list, would be increased further by returning it to the prototype development cycle, as applied to the Standards module.

Formal user evaluation based on user testing continued to provide a reliable safeguard against inappropriate design prior to delivery. The approach of direct observation of the task continued to provide recommendations for design iteration. User testing remained the preferred process to conclude the lifecycle's acceptance phase.

The final lifecycle phase of delivery required both the distribution process and the initial maintenance cycle, (figure 6.8). Small maintenance tasks were revealed by the period of user exploration which followed delivery. Techniques such as scheduled interview or questionnaire distribution were not required as these design problems were reported freely to the developer as they arose.
8.12 Conclusions

The alternative approach to specification and design taken for the Vision module provided a comparative approach to the 'modular system' lifecycle. As confirmed by formal evaluation of the previous modules, the final product of the Vision module development project achieved acceptable levels of functionality and usability leading to its acceptance as a delivered product by the user group. The parallel prototype development process demonstrated the benefit of design exploration to extend the formal user specification leading to user acceptance. Consideration of the module's success in terms of its development cost and usage levels following delivery is presented in chapter 10.

The conclusions drawn at this point in the DAVE development project for the given scenario are presented below.

1. Organisational demands were able to promote product development despite unfavourable assessments of the eventual demand for the product.

2. The information gathered by prototype A's 'module level' specification phase was superseded by design changes arising from user centred prototype evolution.

3. The HUFIT toolset provided a structure for consideration of the factors influencing the product's design. However, the users remained unable to specify the product in sufficient detail without experience of a similar product.

4. The HUFIT tools provided a guide to acceptable performance standards rather than assisting the creative process of design.

5. In cases where the data set is unconfined, the activity of requirements definition can help to guide the design of an appropriate data set structure.

6. Discussion of design issues amongst a panel of three experts provided more relevant recommendations than achieved by a single expert evaluator in previous evaluations.

7. The Molich and Nielsen heuristic list provided appropriate guidance for prototype evaluation by an expert panel.

8. Usability evaluation based on a numerical scale was less effective for this level of design than discussion of features requiring alteration.

9. Generalisation of a user population should only be made when it is too large for the development project to react to the requirements of its individual members.
Formal evaluation confirmed that the Vision model achieved the required level of usability. Through evolutionary development, the functionality of the product exceeded the original specification. Consequently, the 'modular system' lifecycle remained as the accepted approach to module development for the DAVE system.

Chapter 9 discusses the continued evolution of the DAVE system to support changes in the user group’s requirements. This follows the migration of the group’s work programme and the continual advance of technology and its associated user expectations. The processes of maintenance and software support are considered, based on the case studies involving the continued evolution of the Anthropometry and Standards modules.
9 Maintenance and Support

9.1 Chapter Overview

Chapter nine considers the group's changing requirements for the DAVE system and the approach taken to meeting these demands over the course of the project. The task of maintenance is presented in two sections. The first reports on the major modification of modules to alter or expand their function. The second concerns the support for the modules in the form of fault correction, user instruction and hardware maintenance.

The chapter starts with a review of the proposed approach of a separate maintenance phase within the 'modular system' lifecycle to meet changing requirements. The model suggests maintenance as a full time activity, chosen as an alternative to development at the point of 'system level' specification. A process model for this maintenance phase is presented. Maintenance is viewed as a continuation of the evolutionary development process, centring on a return to the prototype development cycle.

The influence of changing requirements, both organisational and technological, on the form and function of the DAVE system modules is illustrated with a discussion of its structural evolution. The changing standards of presentation are reviewed following the systems transition from a single process standalone tool to a multi-process networked application. The lifecycle's response to this drift in requirements is reviewed based on the maintenance of the Anthropometry and Standards modules. The suitability of the chosen processes of re-specification, prototype evolution and delivery and their component activities are considered for module maintenance.

The second section considers the additional maintenance role of computer support in the form of user instruction and fault correction tasks. Support for commercial software and hardware maintenance is reported as a growing demand linked to the spread of the personal computer's influence within the workplace. The chapter provides a revised 'modular system' lifecycle to accommodate this essential support activity.
9.2 The Maintenance Phase

The need to maintain delivered modules was anticipated early in the DAVE project. Fox (1982) reported that the maintenance task could account for more than 50% of the total lifecycle cost for large programs, (section 3.6.5). The traditional lifecycle model chosen as the starting point for the project placed maintenance as a final task. The lifecycle offered a sequential process in which each phase was completed before moving on to the next. The final phase of maintenance cannot be treated in the same way as the preceding phases without assuming that this activity can have an end point. Smith (1991) challenged the assumption that requirements will remain stable throughout the development process. Extending this argument, there is no reason to assume that the users’ requirements stabilise with the delivery of a product. The maintenance phase of this linear lifecycle must therefore be considered as an ongoing activity. The DAVE project’s resource limitation of one person to develop and deliver a series of modules was incompatible with the view of maintenance as a final continuous task. This problem was addressed by the suggestion of a separate lifecycle phase to meet the changing requirements for the system and its components over time, (figure 6.9).

The maintenance phase was proposed as an alternative path to the development cycle. The revised model expanded of the ‘system level’ specification phase to include the option of maintenance along with the possible development projects, (figure 7.7). This model allowed developer effort to be diverted to a maintenance task before returning to the role of software developer.

Figure 9.1 presents the process model, proposed as a guide to the maintenance of the DAVE system products. The model suggested a maintenance cycle based on the continued evolution of the product. The delivery for the DAVE system modules using the same high-level tool which had been used to explore their prototype designs allowed maintenance to be viewed as a continuation of the original design process. By returning to the prototype development cycle, maintenance was able to draw on the experience gained by the developer and user group during the past development projects.

The maintenance model proposed a reduced ‘re-specification’ task at which point changes to the existing design should be specified. Implementation of the specified changes was to be based on the established prototype development cycle. The model omits
the development process's acceptance phase as a measure to reduce the associated time and effort needed to undertake this supporting role. Distribution of the updated module completes the maintenance phase followed by a return to 'system level' specification to select the developer's subsequent work programme.

Figure 9.1 Process Model for Product Maintenance

9.3 The Structural Evolution of the DAVE System

The development project responded to changes in the users' requirements which were linked to gradual technological improvements in the available computer platforms. These design changes can be illustrated by the stepwise design changes made to the system which aimed to enhance the user's ability to access the required system modules.
9.3.1 The Entry Screen

The project proposed the sequential development of a series of modules. The structure of resulting DAVE system was based on the MAPS software which offered a functional model of module access through a central entry screen, (section 5.4.1). The entry screen was delivered alongside the Anthropometry module at the original launch of the system. Its simple design provided an index screen for the DAVE system and access to its contents, (figure 5.7). Its ‘hardware panel’ appearance set the interface standard for the system’s subsequent modules.

The initial structure of the DAVE system was appropriate for its presentation on a dedicated computer platform, located in a central office. The graphical user interface approach allowed by the choice of platform and development tool offered considerable advantages over the alternative of the VAX's terminal used by the MAPS software. As the project progressed, the processing capability of personal computers in relation to their purchase cost improved dramatically. This was reflected by the organisation's shift in policy away from a central typing pool to computer based document preparation by individuals. Resources were made available to purchase the necessary equipment to meet this policy.

The group's introduction to the Macintosh platform through the DAVE project led to its adoption as the group's chosen platform for commercial office applications. This combination of events provided the developer with the unforeseen opportunity to implement the system's modules on the desktop of every group member within the project's time-scale. The developer was able to expand the group's computing capabilities by both supporting the new commercial applications and by introducing a local area network linking the group's computers. This provided the group with central printing and communication facilities to their desks.

The spread of Macintosh computers to the desktop of each group member did not take place until after the completion of the Anthropometry module. Their subsequent networking demanded a re-evaluation of the DAVE system's method of information delivery. The earlier decision to develop for a single dedicated computer was based on a realistic assessment of the limited resources for the project at that time. The
acquisition of more computers from other sources opened new possibilities for presentation, but not without associated costs.

The presentation of the DAVE system on the new machines required the redesign of completed interfaces. The DAVE entry screen and Anthropometry module interface had been designed for presentation on the stand-alone computer's 19 inch monitor. The predominant screen size for the new desktop machines was 14 inches, (section 7.5.3). Transfer to the new standard required redesign of the Entry Screen and Anthropometry module interfaces and a change to the design requirements for future modules. The convenience of desktop access was considered to justify the interface limitation presented by the new design standard.

9.3.2 System Redesign

When choosing a modular approach to development, the DAVE project anticipated multiple versions of the entry screen to accommodate the delivery of new modules. The background graphic approach developed during the design of the first module, (section 6.4.3), reduced the associated development task to simply altering the graphic and copying the necessary functional elements. Similarly, the entry screen's transition to the smaller 14 inch format only required the shrinking of the simple 19 inch interface. Figure 9.2 presents one version of the 14 inch entry screen. The design includes the, by then, standard components of the module buttons, a version control 'identification plate', 'balloon' help, an information window and a simulated liquid crystal display clock.
The decision to provide DAVE across a network of desktop computers altered the way in which the system could be used. The DAVE system became one application among many on each user's computer. The release of a new operating system allowed more than one application to be open at one time. Whilst this is now a common feature of personal computing, it represented a subtle but important change in the use of computers at this time. The convenience of this approach was quickly recognised by the developer and modifications to the DAVE system were made to support this advance.

The adoption of a central entry screen approach required the user to open the 'entry screen' before they could access their target modules. Whilst this was appropriate for a dedicated standalone system, it was an unnecessary added operation within the new multiple application, graphical environment of the desktop computer. The modules were modified to allow them to operate as separate applications which could be opened concurrently. Module icons were presented directly on the operating system's graphical 'desktop' permitting direct access to each module, (figure 9.3). This allowed the group to use the modules in a more sophisticated manner. For example, users were then able to
copy references directly into their word processor. However, the DAVE system’s original entry screen approach was retained as a separate application offering the users the choice to continue to access the DAVE system in a familiar manner.

![Figure 9.3 Customised Desktop Icons for the DAVE System Modules](image)

9.4 Anthropometry Module Upgrade

The decision to provide the DAVE system on the 14 inch screens of the group’s new network was taken during the design of the Standards module, (section 7.5.3). Redesign of the previously delivered Anthropometry module was necessary to allow its wider distribution. This was raised for consideration in the review of the system following the delivery of the Standards module. This work was accepted by the group as a maintenance task to precede the development of the next module. Upgrade of the Anthropometry module became the first implementation of the ‘modular system’ lifecycle’s separate maintenance phase.

9.4.1 Module Re-specification

The Anthropometry module’s original ‘module level’ specification phase guided the users through the specification of functional requirements and data content to produce an initial design from which the final product could evolve. For the maintenance task, the group’s experience in applying the delivered product allowed them to assess its usefulness and recommend changes to its design without the need for carefully guided
elicitation. The goals of requirements definition and data set selection were addressed through the single activity of subject interview.

Whilst the delivered application proved to be robust, improvements were suggested to the functionality, usability and data content of the module. The 're-specification' list for the Anthropometry module is presented below.

- Present on a 14 inch screen whilst retaining the existing functionality.
- Provide access to the full measures database.
- Extend the survey database to the full establishment source.
- Include graphical representation of the data.
- Provide a correct buttock-to-heel photograph representation.

The resulting 're-specification' list, in conjunction with the existing design and components of the delivered module, was sufficient to guide redesign. This removed the role of the function list creation activity which was bypassed.

9.4.2 Module Redesign

It was necessary to balance the proposed functional enhancements with the need to reduce the effective area of the interface. The 19 inch interface could not simply be scaled down as both screens displayed at the same resolution. As a result, two clear design options remained. These were to either split the interface components between two screens, or to design an alternative interface approach.

The demonstrated user preference for photograph based measure searching, (section 6.5.2.3), provided the solution to redesign. A design was prototyped which replaced the measures list with a combined photograph for the selection of measures. This approach left room for the graphical representation of the selected measure's database. The new interface included the ability to toggle between the picture based measure selection and a text based listing of all available measures. The redesigned interface is shown in figure 9.4.
Populations could be added or removed from the module's working database by pressing buttons instead of the check boxes used previously. Selected populations were indicated by the illumination of 'lights' beside its associated button. Data presentation was expanded to show a graphical representation of the data distribution plus values for the range and the key percentiles of 5th, 25th, 50th, 75th and 95th.

User involvement in the design process followed the model of the prototype development cycle, (figure 7.8). They were involved in the review of the design rather than the detailed design of its components. The developer sought general acceptance of the form and function amongst the group before the components represented by the prototype's design were completed and tested.

As the original product was formally tested and passed as acceptable before delivery, (section 6.5), formal acceptance by user testing or expert evaluation was not undertaken for the revised module. The changes made to the original module in terms of functionality and usability were not considered to be sufficient to justify the resources demanded for formal acceptance. Direct user involvement in the module's evolution
provided user control over the changes made during the maintenance phase and an understanding of their consequences on its operation.

9.4.3 Delivery and Support

Providing the new module across the group's network increased the delivery phase's activity of distribution from the simple task of installation on a standalone computer to one requiring greater co-ordination. Separate copies of the DAVE system modules were placed on the hard disks of each of the networked machines. This was preferred to the alternative of running the applications on a central server as this resulted in their slower operation. The cost of this approach was that on the occasions where corrections were made to a component of the DAVE system, it was necessary to distribute the amended copy to all of the networked computers. This task was simplified using a utility which automatically installed the DAVE system software to the correct locations across the group's network.

9.5 Alternative Evolutionary Paths

9.5.1 Expanding the Text Database

The upgrade of the Anthropometry module was overtaken by the development of the Vision module of the DAVE system, (section 8.3). Following the delivery of the Vision module the project returned to the 'modular system' lifecycle's phase of 'system level' specification. This provided an opportunity to consider the development options highlighted during the Standards module project. The module provided a powerful and flexible approach to indexing documents. However, it was recognised that the area of information covered by the module was very limited.

Formal group discussion within the 'system level' specification phase selected the project's continuing work programme. New module concepts such as an administration support module were rejected by the group in favour of a broader implementation of the Standards module's document indexing approach. The information areas specified by the group for support were the group's database of previously applied questionnaires and the growing volume of general computer based documents created by the group.
Evolution of the Standards module to incorporate these new document sources was originally viewed as a single development project. However, it was decided to create separate modules for each category. This approach was adopted to simplify user searching by associating each information area with an identifiable product.

Rather than devising a novel concept, the development of these modules sought to modify of the functionality of the Standards module. Consequently, the maintenance cycle, (figure 9.1), which had been applied to the upgrade of the Anthropometry module was considered to be a suitable model for the General and Questionnaire projects.

9.5.2 The General Module

Design of the DAVE system's General module took the delivered Standards module as a design template. The function list was rejected as an intermediary, relying instead on the code and graphics of the Standards module to provide a starting point for the prototype development cycle.

Specification of the General module matched the 're-specification' activity carried out for maintenance of the Anthropometry module. The results of informal user interview did not raise necessary changes to the recently delivered Standards module. The module's functionality and interface approach were considered by the group to be acceptable. Instead, the need to import miscellaneous computer based documents was highlighted. Any further changes made to the eventual General module arose from requirements emerging from the prototype development cycle. Figure 9.5 illustrates the revised interface design of the General module.
The designer and the user. It serves as a tool to both stimulate requirements of the user population. User requirements tend to change even after incorporating user involvement allows user requirements involvement allows user requirements to develop in parallel. Interaction between the user and the designer (Benjamin, 1969) surface provides the user with an impression of the proposed user, when applied to user testing. Testing, using a representative user population, is central to effective user-centered design. Prototypes allow the trial highlighted knock-on effects.

Figure 9.5 General Module Index Screen

The module was designed to open pre-indexed documents. Common texts such as the establishment telephone list and the stores catalogue were added by the developer. Documents originating from the user group included past reports and fact sheets. Indexing of documents required the user to run the SuperCard utility which had been created by the developer to index the standards documents, (figure 9.5).

Figure 9.6 General Module's Indexing Utility

The General module's design expanded the document selection function from a choice of two standards documents to a selection from a growing list of documents available on the host computer or across the group's network. Users were able to index personal
documents by placing them on their own computer. Alternatively, they could ‘publish’ a document for full group access by placing it on the central ‘server’ computer. The option of viewing graphics was retained from the Standards module despite the added formatting requirement associated with their publication. This was used by the developer to provide a set of stock graphics such as the establishment logo and scanned photographs taken from past experiments, which the group could retrieve across the network.

As reported for the Anthropometry module upgrade, the minor alterations to the Standards module’s functionality to produce the General module were not considered to justify the formal evaluation procedures of the lifecycle’s development loop. Instead, the evolution of the General module’s design relied on informal user feedback during the module’s evolution and following its implementation. This feedback prompted a design change which integrated the ‘indexing’ utility in the General module to automatically search for and index text files.

Delivery of the General module coincided with the change in DAVE structure to direct icon access to the modules, (section 9.3.3). Delivery involved distributing the new module and its representative icon across the network. The module icon was placed on the users ‘desktop’, grouped with the other DAVE icons.

9.5.3 The Questionnaire Module

Although initiated as a single project, development of the Questionnaire module followed the development of the General module. In contrast to the General module’s data set, the source of the Questionnaire module’s data was predominantly paper documents previously applied by the group. Acquisition of the questionnaire database through optical character recognition, was undertaken in parallel with the development of the General module. Completion of the General module allowed development to concentrate on the Questionnaire module.

The Questionnaire project’s characteristics remained unchanged from its template application, the Standards module, in terms of its basic role and the nominated user group. As noted for the General module, (section 9.5.2), ‘re-specification’ through user interview raised changes relating to the data set rather than to the central form
and functionality of the indexing software. With changes relating only to the data set, the requirement for the design intermediary of a function list was replaced by the Standards module as a design template.

The Standards module's indexing approach was retained for the Questionnaire module. This included the two screen structure which split the tasks of searching and display. The design changes made to the template were based on the different characteristics of the questionnaire database. As the documents were solely text based, the photo retrieval elements of the template were removed. The design of the Questionnaire module's database as a single large combined document allowed the interface to be simplified further by removing the document choice function. The questionnaire database was compiled and installed directly as a module resource reducing the time taken to launch the module. The questionnaire database was automatically presented when the module was opened.

As noted for the General module, formal evaluation was considered unwarranted due to the fundamental similarities between the completed modules and its formally evaluated template application, (section 7.6). Whilst usability problems were not encountered following delivery, continued development of the module was instigated to add a particular set of questionnaires to its database in support of a related field trial planned by the group.

9.5.4 Staff Targets

Following the delivery of the General and Questionnaire modules, the project returned to system specification as directed by the 'modular system' lifecycle. System overview did not raise an maintenance tasks, specifying instead another text based information area for support. This additional area was assistance for the generation of Staff Targets documents. These documents defined the requirements of a system as part of the procurement process followed by the group's main customer. The group was responsible for supplying the human factors component of these documents and considered the possibility of applying the DAVE system's document indexing approach to offer potential benefits for the completion of this task.

230
The proposal was to provide support for this new area based on the Standards module template. Selection of this information area led to the return to the maintenance phase. Again, specification confirmed the acceptability of the previously applied approaches to document indexing. Consequently, specification of the Staff Target module through user interview focused on data set definition. This led to the gathering and review of staff target documents previously prepared by the group.

The review of the staff target documents raised concerns over the viability of the proposed module. The first concern was the format of the documents. All of the past documents were available only in a paper form. The poor print quality of many of these archived documents increased the development effort associated with database acquisition, as compared to the General and Questionnaire modules. The second concern lay in the nature of the documents. Due to their commercially sensitive content, copying of the paper documents was not permitted. These problems led to the rejection of the software approach to assist the group when generating these documents. An alternative approach was followed in which a generic document was constructed as a template for human factors input to future staff targets.

9.6 Computer Support

This chapter has concentrated on the main maintenance task of software upgrade. These upgrade tasks required prolonged periods of developer effort concentrating on single maintenance tasks such as module redesign. However, a further maintenance responsibility was placed on the developer over the course of the project. The growth in the number of delivered software products was accompanied by a rise in computer usage within the group. A need for minor design changes and occasional error corrections arose from the group's regular use of the in-house software. The particular environmental characteristic of close contact between the developer and the users led to the demand for immediate attention to user problems. These varied activities are referred to here as computer support tasks. They included error correction, hardware support and user instruction.
Despite the iterative nature of the DAVE development lifecycle a small number of application errors were normally revealed by regular module usage. Whilst the formal evaluations for each of the DAVE modules aimed to provide a representative task to test the module, they could not be exhaustive. It was not possible to anticipate or replicate all the ways in which each module could be used within the acceptance phase.

Delivery of the module removed the constraints of the controlled user test. As each module became the primary information source for a wide variety of enquiries across the group, the opportunities for the users to interact with the system different ways grew. Delivery of the module whilst retaining the close links with the developer bears similarities to the beta testing approach to software testing (Smilowitz et. al. 1994). They noted that the beta test method had the advantage of not being constrained by a highly controlled testing environment. By expecting minor errors to remain in the delivered product the project was able to quickly respond, correcting errors as they arose. Prompt action was important to ensure the continued feedback of the group upon which the survival of the system relied.

The module errors encountered fell into two categories. The first were the result of usage in circumstances which had not been anticipated at the time of development. An example was an instance when the Anthropometry module refused to exit. The developer was on-hand to immediately diagnose and correct the problem. The error was traced to the hardware crash of the server computer during the use of the module. The software refused to exit without writing a log file. The developer was able to simply alter the log code to allow the application to save the file locally if the network was disrupted. The amended log code was copied to all distributed modules and included in all subsequent designs. This problem was revealed by an accident of circumstance which might not have normally occurred in the lifetime of the software. It is probable that other errors remained within the delivered modules which will only be revealed following a precise combination of circumstances. The permutations for testing all possible circumstances prevent definitive evaluation before delivery within the context of in-house development. However, the presence of the developer within the user environment, on-call to maintain occasional errors was effective in minimising the consequences of these dormant errors.
A second form of module error related to the integrity of the large databases which defied complete checks. An example of this problem was the discovery of a gap in the standards database. An oversight during the prolonged document acquisition process omitted a page from one section of the Def. Stan. 00-25 data set. This omission was only revealed by a group member who was using the Standards module on a daily basis for a specific project. Once determined the absent page was acquired and added to the database as a priority task. The possible consequences of this sort of error are more serious for a data retrieval tool. Continued use of these tools depends on user confidence in the integrity of the information provided. Immediate correction of a database gap cannot be expected to restore the full level of user confidence held before such an error. Whilst usage of the Standards module did not noticeably decline after this error, the users were reminded not to accept results without question.

Discovery of errors within the delivered software was potentially serious in its effect on the user group's acceptance of the DAVE system. The occasions of error reporting did not lead to product rejection at any point. The relatively low quality of the commercial software used at the start of the project led to a general expectation of occasional errors. The developer was more frequently required to assist the users following problems with commercial software than with DAVE system software over the course of the project. As the standard of commercial software rose, so did the expectations of the users for in-house and commercial software.

9.6.2 Hardware Maintenance

The level of computer usage within the group grew in line with the developing DAVE project. Whilst there was a partial link to the DAVE project, the group's increased use of computers owed equally to a change in the organisational culture towards the use of computers for office based tasks. A consequence of this trend was an increased demand for maintenance. As a result of the relative levels of computer experience, the role of hardware support fell to the developer. The implication of this added task on the DAVE project and its associated lifecycle was the occasional diversion of the developer away from the set work programme. The group's increasing dependence on office based applications made computer support a priority overriding all other work programme commitments. This was an organisational demand which was not present at the start of
the project. Its influence on the project increased to become an important demand on the project's resources towards the end of the reported time frame.

9.6.3 User Instruction

The requirement for user instruction for the DAVE module was minimal. The simplicity of the modules' design and user centred approach to ensuring usability allowed group members to access the complete functionality of the modules without assistance. Enquiries relating to the DAVE system related mostly to the transfer of information from modules to other applications such as a word processor for report generation.

In contrast, the variety and complexity of commercial applications acquired by the group grew over the course of the project making this the main source of group enquiry. As noted for hardware support, the developer became the source of instruction for both the in-house products and this increasing range of commercial software. It was notable that despite the rise in functionality offered by commercial applications, such as the group's word processor, the group members tended to continue to use the same familiar functions. As noted for the demand for hardware support, the influence of this time demand on the DAVE project and its lifecycle increased over the reported time-frame.

9.7 Discussion of Methods

This chapter has taken a longitudinal view of the project. It reported the structural evolution of the DAVE system, the upgrade of its components and the organisational drift towards increased computer utilisation. The task of supporting these changes was met within the resources of the DAVE project under the control of the user group. Each upgrade task represented a change in the group's expectations for the computer/software package.

Observations of the group's input to the design of the DAVE system and its acceptance tie in with the suggestion that users require experience of a similar system on which to base their understanding of a subsequent product's requirements (Hekmatpour and Ince 1987, Luqi 1989). The group's original concept of requirements had been limited by the absence of past experience of similar systems. The prototyping approach allowed
them to develop a concept of what was achievable as a benchmark for their expectations. As the project progressed, the users' judgement for both commercial systems and in-house software showed links to their experience of these products. The gradual drift in the group's requirements can be explained as their response to organisational changes and the rising standards of the computer industry.

The 'modular system' lifecycle of the DAVE project responded to these demands formally through its maintenance phase and informally through a computer support activity. The procedural approach to maintenance was based on the continuation of the module's original development sequence. Development and delivery of the DAVE system modules using the same high-level programming tool allowed simple modification of applications as the code and interface elements were easily accessed for redesign.

9.7.1 Module Re-Specification

The Anthropometry module's maintenance task arose from the change in the interface design standard set during the development of the second DAVE system module. In the Anthropometry module's original design, the equivalent phase of 'module level' specification was used to refine the concept produced by the broader 'system level' specification. Similarly, module 're-specification' aimed to define proposed changes in more detail. The 're-specification' process provided an opportunity to seek group feedback based on their experiences with the delivered product.

User interview raised other possible improvements beyond the original aim of providing the module across the new network. Many of these features had not been previously discussed with the developer, highlighting the importance of a point in the cycle at which there is formal consultation of the users. Although increasing the maintenance task, the group took the view that the maintenance phase should address the additional features raised.

The proposed changes represented small improvements in functionality and usability rather than the correction of errors. The project lifecycle would not have allowed the diversion of developer effort from the main project to implement these improvements. Whilst user instruction and the correction of errors was provided on request, large maintenance tasks and design improvements were not undertaken immediately. Instead,
they were referred to the system specification phase to avoid unnecessary diversion of
development effort from the current project.

The maintenance phase's 're-specification' task differed considerably from the
specification task for the initial prototype. When specifying the original module it had
been necessary to guide the users in determining the appropriate module functions or
database content for the module. Its eventual specification was only able to provide a
loose guide to design which was evolved towards their requirements following the
prototype development cycle. In contrast, the experience of the users in conjunction
with the existence of a module design allowed direct discussion of a range of possible
improvements to the module. The users did not require separate guidance to consider
module functionality, interface design or the data set content. They were able to present
informed views covering all of these issues when discussing possible improvements to
the module.

The Staff Targets project demonstrated an alternative role for the 're-specification'
process as a check against inappropriate development. The detailed consideration of the
issues surrounding the development of a module within the suggested information area
raised and confirmed flaws in the concept which prevented the project from proceeding
to the next stage of development. The 're-specification' process allowed the associated
risks to be recognised at an early stage, diverting effort to a more acceptable technique
of supporting the group's work programme.

9.7.2 Continued Evolution

The lists produced by the 're-specification' process provided design goals in terms of
functionality and content. For the first upgrade task, the main goal was to reduce the
interface area of the Anthropometry module. Meeting this requirement involved creative
redesign to balance the limitations of space with an extension of its functionality.
Experience of the group's interaction with the delivered module allowed this balance to
be achieved by rejecting text based selection in favour of the graphical interaction
preferred by the group. Text based selection was included for consistency but remained
little used by the group. The new design evolved to support the enhancements suggested
by the user group. Fitting these features within the new interface relied on the
flexibility of the prototyping tool to explore design alternatives. Various design options
were tried and rejected before an approach acceptable to both the developer and group was achieved. Any attempt to formalise the design sequence beyond the guide of the prototype development cycle would have placed unworkable constraints on what Bott (1988) considered to be an essentially creative process.

Although continued evolution appeared to lead to an acceptable design for each of the reported cases, this approach continued to present risks. An important concern was process's reliance on the contribution of all members of the user group. Whilst the formal interview approach ensured even consultation, the level of contribution varied between members. It was necessary for the developer to judge each suggestion for modification against the associated impact on the module in terms of usability and functionality in addition to the development effort required to make the change. Any request considered to be controversial was discussed further with the individual and raised with the rest of the group before being accepted for redevelopment.

Another critical feature of the design process was the eventual product's dependence on the ability of the developer to devise design solutions throughout the process. The final product relied on the individual aptitude of the chosen developer in creating solutions within the limitations of the development tool. The 'modular system' lifecycle could not achieve an appropriate design in the absence of the fundamental development skills of design and development. This is a problem common to any form of novel software design. Attempts to tightly define a prescriptive design formula risk constraining the creative process which is central to novel design. Although the quality of the eventual design necessarily depends on the skill and experience of the designer, formulaic design does not offer and acceptable alternative to individual creativity. By ensuring the integral involvement of users and experts, the 'modular system' lifecycle guides the developer in drawing on the creativity and insight of others.

The prototype development cycle provides a suitable framework for managing the potentially chaotic process of creative design. It provided support by separating product design into a series of smaller element tasks. This allowed the developer to concentrate on manageable sections of the product, whilst retaining a concept of the product as a whole through regular product level reviews. The cycle always returned to an overall assessment ensuring developer and user consideration of the evolving design.
9.7.3 Informal Acceptance

The 'modular system' lifecycle relied on user involvement to both guide the design towards an acceptable form and to encourage continued project support by reinforcing the group members' stake in the product. User group involvement throughout this exploratory stage was considered to contribute to the acceptance of each upgrade without challenge. Although effective for the DAVE project, this approach relied on a close working relationship and the avoidance of personal conflict. These project characteristics grew from the particular environment within which the DAVE system was developed and the personalities of those involved. Achieving this level of informal involvement could present difficulties within an alternative organisational environment or with a different mix of personalities. However, the user centred framework provides a strategy to counteract the environment's risks with its potential advantages.

Whilst the success of the original module development approaches can be considered on the basis of the results of their formal acceptance phases, similar assessment of the maintenance phase processes is prevented by the absence of formal evaluation of the revised modules. The decision not to include a formal acceptance phase before delivery was based on two factors. Firstly, the basic functions remained the same for the revised modules with any changes tried and accepted by the user group. Secondly, project pressures required a quick return to the visibly productive process of new module development. The group's informal expression of satisfaction with these products led to the initial conclusion that each product was successful in terms of user acceptance. Objective evaluation of the success of these modules is considered further in chapter 10.

9.7.4 Development Through Redesign

The development of the Anthropometry, Standards and Vision modules represented novel designs, each requiring exploration to arrive at an effective approach for presenting a particular type of information. Each module required a detailed specification phase and the safeguard of formal evaluation procedures due to a lack of precedent. In contrast, the development projects for the General and Questionnaire modules were able to copy the solutions produced and validated by the design and evaluation of the Standards module.
The 're-specification' processes did not raise design issues beyond definition of the database for either of these Standards module variants. Whilst interface changes did emerge during the development cycle for both modules, they responded to the functional requirements demanded by the data. The database for the Questionnaire module existed as a single part allowing the module’s design to remove the user task of document selection. In comparison, selection from multiple data sources was necessary for the General module. It was not necessary for the 're-specification' procedure to highlight these requirements as the development process both revealed these changes and provided a method for testing design solutions.

9.7.5 Requirements Drift

The 'requirements drift' refers to a rise in the expectations of the group linked to the gradual rise in commercial software and hardware standards. Computer technology has continued its evolution from its origin as a specialist calculating device into a flexible tool which is accessible to a wider population. In the past, traditional lifecycles have been able to concentrate on the specification of a product's functionality, limited to a few interface alternatives. Although apparently slow to respond, the computer industry has raised the standards of usability to attract a new market. Continual advances in computer hardware has allowed the industry to pursue new approaches for the user to interact with the computer and direct its processing capabilities. Inevitably, these trends influence the commercial software market and in turn, the potential customers for in-house software projects.

Figure 9.7 illustrates the main phases in the evolution of the DAVE system's structure. The basic interface style and core code developed for the entry screen and the anthropometry module at the start of the project were retained for the rest of the DAVE system products. Design and development of these features involved a high rate of change over the initial period of the project through iterative prototyping. The result was a standalone product running a single application presented on a 19 inch colour screen.

The model proposes that at the start of the project the group's experience with simple text based applications set their initial requirements and associated expectations. The group were initially unable to set specific requirements. The development of test applications and the first DAVE module provided examples on which they were able to
align their concept of the scope of in-house software. The new graphical interface approach of the chosen development tools, as demonstrated by the early prototype, raised their expectations of the product. Meanwhile, the iterative addition of functions and the refinement of the interface allowed the design to exceed the requirements level at which the product might be considered acceptable for delivery.

![Figure 9.7 DAVE System Evolution to Meet Changing Requirements](image)

The spread of computers within the group changed the way they were used from a central resource, accessed occasionally, to a desktop tool used for a wide range of applications. The next step was the linking of the group's desktop computers to provide an internal network. The group's support for modification of the system to allow delivery across this network provides evidence of the gradual change in product expectation. This marks an incremental change in the system's design. It is represented in figure 9.7 as a curve to indicate an initial large design change followed by refinement.

The final major structural change to the DAVE system was its support for opening multiple applications. An advance in the commercial operating system made it possible to provide this new feature. Adding support for this new operating system standard to the DAVE system required both the determination of a design solution and its implementation within all DAVE system modules, delivered and future.

Each of the mid-life upgrades reported refer to changes in the way the group were able to access the DAVE system module. These changes did not increase the functionality of the
system or the basic usability of each module. Instead, the changes increased the convenience of accessing the software. Firstly, the users were allowed to launch the system without leaving their office. Subsequently it became possible for them to launch each module directly without closing their current application. Neither of these features could have been included in the original specification of the system as they only became technically possible as the project progressed. Future changes to the DAVE system might include porting the software to an additional platform such as a hand held personal digital assistant (PDA) or expanding the interface to include voice input/output.

The users' acceptance of these commercial trends is represented in figure 9.7 as a gradual increase in user requirements. The maintenance tasks were undertaken in response to user views. The group member's exposure to new software approaches gradually raised this requirement level. It was necessary for the DAVE system to evolve in line with these software trends to remain acceptable to the group. Falling below the standards of the group's other software products risked eventual neglect of the DAVE system modules.

An indication of this gradual change in expectations was provided by the allocation of fault. Inexperienced group members tended to attribute difficulties when using software to their own abilities as opposed to the software's design. The group's growing experience and confidence in using computers led to individuals reapportioning blame, more appropriately, to the software. The user centred DAVE development approach encouraged the group to highlight usability problems in the software for correction. Without a means of correcting the faults of commercial software the group simply voiced complaints about inappropriate features amongst themselves.

Figure 9.8 presents a similar representation of the Anthropometry module's evolution in terms of the changes to its design over time. Again, a dotted line is used to represent the user requirements. The level of functionality and usability offered by the initial prototype started below the requirements of the users. As the prototype evolved, it became more suitable for implementation. Through user centred design within the prototype development cycle, the design achieved and then exceeded the requirements of the group. Whilst figure 9.8 defines points at which a requirement for change was accepted and implemented, it was not possible to isolate the points at which the group members recognised possible improvements to the module. Consequently, this model
presents the change in requirements as a gradual rise. Use of the module continued throughout the project indicating that it continued to meet the group's criteria for acceptance, despite rising commercial software standards for functionality, usability and convenience.

Figure 9.8 Anthropometry Module Evolution to Meet Changing Requirements

The ability of the lifecycle to support the occasional upgrade of delivered modules depended on the flexibility of the prototyping environment. The accessibility of the programming code and graphic components of each module allowed simple alteration of the design. As the requirements for redesign arose primarily in the interface approach, most of the Anthropometry module's functional code was transferred to the new design intact.

9.7.6 Computer support

This chapter makes a distinction between the task of maintenance and the task of computer support. Following the 'modular system' lifecycle, 'system level' specification directed developer effort towards the upgrade of both the Anthropometry module and the Standards module. Maintenance of the Anthropometry module brought it into line with the DAVE systems evolving presentation standard. Maintenance of the Standards module extended its contribution to the group's work programme by expanding its data set. In each case the work required to complete these maintenance
tasks required a prolonged period of developer effort, representing separate projects in
their own right. As the project progressed a lower level demand emerged for computer
support.

Computer support tasks differed from the formal maintenance tasks in several ways.
Firstly, the support tasks required relatively little developer effort to complete. User
instruction, code correction or hardware maintenance could normally be completed
within an hour. In comparison, the reported maintenance upgrade tasks required
developer effort over a period of weeks or months. Secondly, computer support tasks
generally arose whilst the user was trying to use the system as part of their work
programme. In order to complete their task it was often essential that instruction or
fault correction be provided immediately. The larger maintenance tasks were less
urgent. The proposals arose from the use and consideration of the system over a
prolonged period, and were able to wait until the lifecycle’s ‘system level’ specification
phase to be chosen. Finally, computer support tasks were more frequent. The wide range
of support tasks could be set by any member of the user group at any time and were
addressed immediately. This random activity did not fit within the otherwise sequential
process guided by the ‘modular system’ lifecycle. However, as the purpose of the DAVE
system was to support the group, it was essential that any barriers to its use be cleared
whenever they arose.

9.7.7 Lifecycle Redesign

Figure 9.9 presents a revised ‘modular system’ lifecycle model. The new model included
a computer support phase to accommodate the irregular support tasks highlighted in the
DAVE project. Computer support is presented as a ‘floating’ phase. The demand for
immediate instruction and fault correction prevents the firm linkage of this phase to the
rest of the lifecycle. It is included as a ‘module level’ phase as each of the DAVE specific
enquiries related to the use of a specific module as opposed to the system as a whole. By
separating this phase from the lifecycle sequence the model represents the ability of the
computer support tasks to interrupt the current project’s development or maintenance
sequence within the in-house environment. Whilst this diverts developer effort from
the formal lifecycle, anything but immediate support of delivered products was
considered to risk user rejection.
Figure 9.9 Maintenance and Support within the 'Modular System' Lifecycle

The computer support role started with the delivery of the first DAVE system module, increasing with the subsequent release of other in-house products. However, the role of DAVE system computer support was quickly overtaken by the task of supporting the group's hardware and commercial software. This effect was heightened by the organisational shift towards computer use of office tasks over the course of the project.
Development of an in-house capability for software development brought with it an expansion in the group's use of computers as a normal aspect of their work programme. The rising number of in-house products delivered to the group inevitably increased their requirement for maintenance. The 'modular system' lifecycle's approach to a formal maintenance phase allowed the limited development resources of the group to support effective upgrade of the delivered system. At a lower level, the computer support roles of user instruction and fault correction, both software and hardware, placed increasing demands on the project's resources. The growth in the group's use of commercial applications and the support for these products produced the greatest conflict with the continued in-house development.

The conclusions drawn from this review of the task of maintenance and computer support for the given scenario are presented below.

1. Delivery of a product does not mark the end of development as the permutations of use in the working environment cannot be fully explored through formal user testing.
2. Creative design based on individual skills forms an essential component of the prototype development cycle.
3. The project's in-house products were required to respond to a drift in requirements related to improvements in the standards of the commercial software industry.
4. Module delivery using a high-level development tool allowed software upgrade to continue the product's evolutionary design process.
5. Past experience of the product allowed the user group to competently re-specify the product without the specific guidance offered by the 'module level' specification activities.
6. The detailed consideration required by the 're-specification' process provided a control against inappropriate development.
7. User centred modification and the retention of the template application's core functionality led to acceptable designs despite the omission of formal evaluation.
8. Support for in-house products in the form of user instruction and fault correction must be made available whenever the demand arises.
9. The accessibility of the code and interface components of high-level software supported immediate maintenance.

10. Developer effort was diverted from the development project by the requirements for commercial software and hardware support.
10 In-House Software Usage and Development Cost

10.1 Chapter Overview

Chapter ten assesses the relative success of the in-house software products reported in this study. Background logging code was a standard component of each delivered in-house application. The resulting log files provided a measure of each application's use following their delivery on which an assessment of their value could be made. The study's assessment of the cost associated with each in-house product was based on a measure of allocated developer time throughout the DAVE project.

The data gathered from the usage logs are presented in two figures. The first separates and presents the number of launches per month for each module of the DAVE system. The trends and fluctuations in the use of each module are considered in relation to the other delivered in-house products. The second figure presents the usage results in the three categories of DAVE system modules, supplementary applications and total in-house software products. These results distinguish the usage patterns of the information retrieval modules of the DAVE system and supplementary applications produced during the project.

The time allocation data was drawn from a work log maintained by the DAVE system's developer during the project. The data are presented for each month, as a percentage of the total recorded development effort for that period. Trends in work programme tasking are illustrated by a banding approach to the presentation of the table. The categorisation of developer effort is outlined and the limitations of this data source are discussed. Variations in the development effort reported for each project are considered in relation to the influences of the project, the organisation and the user group.

The project's objective measures of value and cost are considered for each in-house product to assess their relative success. The influence of the alternative lifecycle procedures applied to development on each module's success is discussed.
10.2 Background Usage Log

10.2.1 Data Gathering Technique

The technique of background logging was originally applied to the formal user evaluation of the final Anthropometry prototype, (section 6.5.2.2). The prototype recorded the subject’s sequence of actions for later analysis. Although this technique was rejected in favour of direct observation for future formal user evaluations, the logging code was included as a standard component in all future designs as a potential aid to evaluation and maintenance.

The first release of the DAVE system recorded a log file to the hard disk of the standalone computer. The code was modified to record the log file to a central computer with the move to a network based system. This both simplified file collection and prevented the accidental deletion of log files by the group.

The log code for the supplementary applications, (appendix H), recorded additional data depending on the function of the application. The Lab application logs included interaction data such as reaction times for analysis after the experiment. The Map Simulation stored button interaction and vehicle position in a form that allowed experimental runs to be replayed on the map at a later date. Whilst this data was valuable in meeting the experimental aims of the software, it is not presented here. Instead, the basic measure of application launches is used. This measure is consistent for all of the application logs, allowing direct comparison. Figures 10.1 and 10.2 present the frequency of use for each reported, in-house product. The data shows the number of times each application was launched per month over the period of the DAVE development project. It should be noted that these figures use lines for presentation purposes and that a linear link between the data points is not implied.
10.2.2 Limitations of the Usage Data

Whilst the results provide an important objective measure of product acceptance, limitations of the data must be considered when drawing conclusions. The data does not:

- Distinguish between the individual users of the software.

The data logging routine did not demand the user's name as this added task was considered to be inappropriate. Consequently, there is no record of the number of accesses for individual users. However, certain peaks of usage can be attributed to an individual working on a specific project. The peak in usage of the Standards module, April/May 1994, can be attributed to a standards comparison project.

There were several changes to the composition of the group over the course of the project, including an annual turnover of student members of the group. Two members of the group left the organisation and were replaced by new employees at the same level. Whilst periodically fluctuating, the size and skill base of the group remained similar for the full period reported by this study.

- Indicate the user's level of success in locating the required information.

As use of all DAVE software was voluntary, it is assumed that repeated failure by users to find the required information would have led to either feedback to the developer or rejection of that particular module. In turn, this would have been reflected in the usage data. With the exception of the Vision module, this was not encountered for the DAVE system software. The cause of low Vision module utilisation is discussed later.

- Show how long the user interacted with the software.

Although the interaction log recorded the period each application was open, the applications were not necessarily closed by the individual after use. The original DAVE system required the user to close a module before opening another application. The move to a new operating system allowed multiple applications to be open at the same time, removing the need to close an application after use. Consequently, the time an
application was open was considered to provide a misleading impression of the total usage of the software.

This change in the operating approach must be considered when comparing the launch frequency for applications. The new operating system allowed users to keep the module open in the background allowing them to return to the module at a later point without triggering a separate log file. Consequently, usage levels for this later period may be slightly reduced in comparison to use under the original operating system.

10.3 DAVE Module Usage

Figure 10.1 presents the number of times each of the DAVE system modules was launched for each month of the DAVE project. The data was taken from the log files recorded by each module after its official delivery. Launches of the evolving prototype for informal evaluation has not been included in the 'usage' data set as it does not distinguish between user and developer interaction. Figure 10.1 illustrates the voluntary usage of each of the DAVE system modules by the target user group.

Launches of the DAVE system entry screen are not represented separately. It remained the sole point of access for the system until the introduction of direct access with the General module, (section 9.5.2), after which use of the entry screen declined.

Formal evaluation of the DAVE system modules showed that the delivered modules achieved their functional goals by providing the specified information in a form accessible to the set user population. However, long term usage levels allow us to judge the overriding goal of achieving voluntary usage by the target population. It is these levels of usage which provide the final assessment of module success.
10.3.1 Anthropometry Module Usage

The Anthropometry module was officially delivered in September 1992. The relatively high number of launches in this first month relates to user group exploration of the system. Despite involvement in the development of the tool, the group members demonstrated an interest in the operation of the module. Few of these Anthropometry module sessions can be attributed to retrieving data in support of the group's work programme.

The usage levels quickly dropped to under thirty launches per month. Whilst this is considered to be a more representative requirement for anthropometric data within the group, this drop coincides with the Map Simulation field trial, (appendix H). Preparation, completion and reporting of the trial diverted the work programme of most members of the group away from tasks requiring the retrieval of anthropometric data. The usage level rises again in October 1993, linked to a specific crewstation design project.

A resurgence in use of the Anthropometry module is seen during March 1994 following its transfer to a 14 inch screen format and its distribution across the group's new network. By placing the module on the recently acquired desktop computers of each member of the group it was anticipated that usage levels would rise. It was suggested that the previously moderate levels of use had been an affect of 'user inertia' in which the need to physically move to a separate office was a demotivator for use. Although levels rise initially, Anthropometry module usage fell back to a similar level observed for the standalone application. Whilst placing the application on each user's desktop made it more accessible, it did not alter the amount of work for which the group required anthropometric data. However, it is considered that the suggested drift in user expectations, (section 9.7.5), would have led to the premature rejection of the module if the reported design changes had not been made.

The Anthropometry module's upgrade halted a trend which may have resulted in the eventual neglect of the product. Before the module's upgrade, usage levels showed a decline. The move to a networked application reversed this decline demonstrating the benefit of mid-life upgrade through the lifecycle's maintenance phase. However,
consideration of usage levels beyond this upgrade show the raised usage levels to be temporary. The remaining period of the project shows a continuation of the downward trend. Whilst this trend promised the eventual rejection of the application, the Anthropometry module remained the sole source of detailed anthropometric data for the group, thus preventing complete neglect.

A contributory factor to the declining use of the Anthropometry module may have been the relevance of the module’s database. The data contained by the module had been gathered more than twenty years previously. Whilst this remained the only source of anthropometric data specifically gathered for the given populations, trends in the wider population and changes to selection policies for the groups surveyed continued to reduce the reliability of the data. Whilst, the design of the Anthropometry module allowed its database to be updated, the data required to upgrade the module further depended on a new population survey. This data was not forthcoming within the study’s time-frame.

10.3.2 Standards Module Usage

Delivery of the Standards module had a major impact on the overall usage of the DAVE system. By providing on-line access to the key reference documents of the group, the module encompassed many of the information areas highlighted in the group’s original concept of a vehicle ergonomics specific information system. This broad database increased the occasions on which the Standards module could be applied to the group’s regular work programme. This is supported by figure 10.1 in which its level of usage can be seen to have been consistently higher than the other modules of the DAVE system.

As observed for the Anthropometry module, the initial level of usage for the Standards module was relatively high. Again, this period can be attributed to users exploring the system. Despite user involvement in the design process, curiosity in the limits of the application prompted product exploration over this post delivery period. Unlike the other modules, usage of the Standards module shows a peak exceeding the initial levels of product exploration several months after delivery. This period over April and May 1994 relates to a specific work programme in which the group’s main reference documents, contained within the Standards module, were compared against other available reference documents. Despite the higher overall levels of usage achieved by
the Standards module a gradual decline appeared, similar to the decline noted for the Anthropometry module.

The Standards module was developed as a more efficient alternative to the paper document. In this role, it was presented as a possible replacement to paper, allowing fine searches of large documents. Observation of the group's use of the Standards module revealed a variety of approaches to applying the module. In a couple cases, the individual's preference for the paper format overcame their motivation to make any use of the on-line resource. Of the other group members, only two used the Standards module as their sole reference source. The remainder tended to use both media together. The Standards module could be used to locate references not found by quickly skimming through the paper source. Once located, the users would frequently refer back to the paper document to read the reference in detail. The ability of the module to copy text and graphics directly to a report document was used infrequently.

10.3.3 Vision Module Usage

Whilst the data gathering technique developed by the Vision project was adopted as the group's standard for vision assessment, the delivered module failed to make an impact on the work of the group. The Vision project followed a period in which vision assessment reached a peak, as part of the group's work programme. However, after the delivery of the software, there followed eight months in which there was no work programme requirement for vehicle vision assessment. This absence of applicable tasks prevented the immediate implementation of the delivered product.

By the time a vision assessment task arose, user confidence in applying the software in the field was not high enough to use the module directly. A major concern lay in the battery life of the portable computer available to run the software. The eventual assessment used the new technique, but recorded the data to paper. The data was then transferred to the Vision module on a desktop computer in the group's offices to visually confirm the integrity of the data. No further vision assessment was undertaken by the group in the study's time-frame. The software remained available on the group's desktop and portable computers leading to software demonstrations and user
exploration. These instances account for the occasional launches of the software over this period.

Whilst the vision module project successfully formalised an improved assessment technique, the software product failed to meet the DAVE project's main objective of supporting the group's regular work programme through voluntary use. This does not appear to be linked with the functionality of the software structure or interface design, which were both shown to be acceptable to the users, (section 8.8). The neglect of the delivered product resulted from an absence in requirement for the tool arising from a change in the group's regular work programme. Consequently, criticism of the development approach must be focused on either its failure to anticipate this change in requirements or its incorrect assessment of the implications of the low levels of predicted usage. The implications on the DAVE project's approach to project selection is considered further in section 10.6.1.

10.3.4 General Module Usage

The record of usage for the General module shows a peak following delivery. This is consistent with the usage peaks seen to coincide with the delivery of the previously delivered modules. As this product was delivered near the end of the month, the initial usage peak was spread over two months. Following this early surge of interest, the level can be seen to drop to a lower, more consistent level.

The comparative success of the Standards module suggests that the user tested functionality and interface adopted by each of the text retrieval modules produced a tool acceptable to the user group. Whilst the General module is not considered to have failed to support the work programme in the way seen for the Vision module, its full potential for the support of the group was not reached. Its actual usage level failed to meet the expectations held for the product.

By allowing users to index their own documents the General module offered the possibility of on-line access to a growing range of indexed documents. Although the module settled to a stable level of usage, the indexing facility was only used occasionally. Instead, the group tended to use only the pre-indexed documents provided by the
developer on the central server computer. The module’s potential depended on the growth of the user generated database. Failure of the user group to use document indexing for their own documents limited the range of information served by the module. The higher levels of use expected for the module depended on the module being able to access an ever broadening group of databases. Despite the lack of user indexed document’s, the General module continued to be used consistently following delivery.

The incorporation of the indexing utility within the General module was carried out as a computer support task in response to the group’s reluctance to index their own documents. By automatically indexing documents placed in the General module’s database folder, the aim was to boost module usage by broadening the available databases. This user specified modification removed the need for users to launch a separate utility in order to index their own documents. However, this facility was only used by a couple members of the user group. This reveals a disparity between the user group’s perceptions of their requirements and their actual needs.

10.3.5 Questionnaire Module Usage

The Questionnaire module was completed and delivered shortly after the delivery of the General module. The delay in delivery related to the added effort required to gather the questionnaire database. Although present, the common ‘exploration’ peak of usage was lower for the Questionnaire module than seen for the Standards or General modules. The interface similarities and the narrower database was considered to reduce the incentive to explore the software in comparison with the previously released text retrieval modules.

The use of the Questionnaire module fell to a lower level than recorded for the other text retrieval modules. As the range of information contained by the General module was wider than the questionnaire references, opportunities for its application were more frequent. Use of the Questionnaire module shows wide fluctuations with occasional peaks of use. The peaks seen in October 1994 and May 1995 relate to preparation for specific field trials. These trials required new questionnaires to be generated, for which the previously applied questions offered a starting point. The second, higher peak followed a
period of developer activity in which the Questionnaire database was extended to include two past questionnaires relating specifically to the approaching field trial.

The Questionnaire module was similar to the Vision module in that the predicted usage of the tool was low and occasional. However, the Questionnaire module presented a lower risk as the continued evolution approach to development required less developer effort than the full development cycle. The usage record shows that the module achieved expected levels of usage. A contribution to the relative success of the Questionnaire module was the requirement to generate a new questionnaire soon after the module’s delivery. It was noted for the Vision module that its neglect was linked to an absence of related work following delivery. The use of the Questionnaire module shortly after delivery provided a precedent, establishing sufficient user confidence to apply the tool after a delay.

10.4 In-House Software Usage

Figure 10.2 presents the total number of launches for software developed over the period of the DAVE development project. The data is grouped into the categories of DAVE module usage, supplementary application usage and total in-house software usage. The data is presented as the number of launches per month over the course of the DAVE project. The data for each group starts with the delivery of the first application, in line with figure 10.1.
Figure 10.2 Total In-House Software Usage
10.4.1 Total DAVE Module Usage

From an initial peak following the launch of the Anthropometry module, the use of the DAVE system software shows a lower, stabilised level of usage until the launch of the Standards module. This was the first of several modules launched within a period of eight months. The increase in modules available to the group led to a peak of DAVE system use in May 1994. Whilst this peak relates predominantly to the demand for standards information, the total level of DAVE system use remained high for the next six months.

The usage data from the individual DAVE modules, (figure 10.1), shows both a wide range of usage between modules and also considerable fluctuation in the usage levels of individual modules. The Standards module accounts for the largest proportion of launches over the project time frame with consistently higher utilisation than the other modules. In contrast, the Vision module makes the smallest contribution to the total level dropping to near zero usage soon after delivery. The results show that the remaining modules achieved stable levels of voluntary use with occasional peaks relating to the group’s work in the specific area supported by the module.

The usage records for the Anthropometry and Standards modules showed periods of gradual decline over the course of the project. This is reflected in the use of the DAVE system as a whole. Despite this decline the cumulative level of DAVE system software usage did not fall below 50 launches per month. The downward trend relating to the Anthropometry module was reversed by the introduction of the Standards module early in the project. Usage was further boosted by the upgrade of the Anthropometry module at a later date. The overall level of use was maintained at a consistently high level over the following period by the regular release of new modules. However, this was followed by a period of relatively little change in the DAVE system which was accompanied by a return to the trend of a gradual decline in use. Whilst the DAVE system modules provided the same level of support over this later period, changes in the group’s work programme, the organisational environment and the users’ expectation of software combined to reduce module use in support of the work of the group.
10.4.2 Total Supplementary Application Usage

The combined launches of the Panel Prototyper, the Map Simulation and the Lab Map tool, (appendix H), are presented in figure 10.2 to illustrate their contribution to the group’s work programme. The data logs recorded by these applications were important as a tool for gathering a detailed record of user interaction in support of the group’s research programme. However, the logs provide little information concerning their intended use.

Figure 10.2 shows usage peaks which appear as a consequence of their controlled implementation. These peaks do not indicate the level of success in relation to the other applications as the number of times each supplementary application was applied depended on the experimental design or tutorial plan. In comparison, usage levels for each of the DAVE modules represents an individual’s free choice to launch an application, indicating their perceived value of each module over time. The determination of success for the supplementary applications must rely instead on the achievement of the specific goals for which each application was developed. As each of the experimental support applications was fully implemented, leading to published results, they were considered to be successful.

The supplementary application usage is presented to illustrate an additional role of this software, rather than as a measure of their success. Each of the supplementary applications was developed to meet the requirements of a single project. Use of these applications after this point was not anticipated. After development, each of these applications was freely available across the group’s network and could be run from their desktop computers. Although intended for a single purpose, they continued to be used beyond each product’s short period of intended use. This additional low level use was voluntary and therefore reveals an added value provided by the applications in support of the work of the group.

Despite the intention to discard the supplementary software following implementation, figure 10.2 shows a continuation of low level use throughout the project. This continued use raises the assessed contribution made by supplementary applications to the work of the DAVE user group and to the design of the DAVE system. Although the first release of
the Panel Prototyper included usability problems, it remained a popular tool, influencing the design of all subsequent software developed during the project. The code, interface components and application structure were refined and carried across to each subsequent project. Similarly, the Map Simulation led to the design and implementation of the Lab Map tool to further contribute to the group's research programme. Each of these applications became an important aspect of the group's research capability, becoming a regular feature in presentations to visiting groups.

10.4.3 Total In-House Software Usage

Figure 10.2 presents the grouped data for both the DAVE modules and the supplementary applications. The results confirm the DAVE system to be the dominant product in terms of voluntary use by the group. Initially, the quickly developed supplementary applications accounted for all of the delivered in-house products, dominating the usage chart. However, these high levels of use were temporary as their main function ceased at the end of the task for which they were designed. In comparison, the DAVE system modules continued to play a larger role in the regular work of the group after delivery.

To consider the relative contribution of these different classes of in-house product it is necessary to look beyond usage levels and consider their impact on the group's approach to their work programme. Whilst use of the DAVE modules represented an alternative approach to retrieving information, these new search techniques did not replace existing information sources. The continued use of the DAVE system modules cannot be taken as proof of an improvement in the effectiveness of the group as a direct result of the software products. However, it does confirm a group perception of benefit from applying the various modules of the DAVE system. Without this perception the usage levels of the DAVE system would have fallen to zero.

The development of supplementary applications exceeded the goals of the DAVE system. The visible contribution made by these applications to the work of the group, in some respects, outweighed the effect of the DAVE system software. The supplementary applications provided tools which allowed the group to extend their approach to research and instruction. The new approach of computer based simulation became a key tool for several of the group's research programmes, changing the way in which certain
research questions were investigated. In comparison, the DAVE system software had a broader but less distinct impact on the group's work. The delivered modules provided more efficient methods of finding information required for the group's regular work programme without fundamentally altering the way this information was used. The sustained levels of use indicate a perception of benefit within the group which supports the conclusion that the system achieved its original aim of supporting the work of the user group.

An additional contribution made by the DAVE project to the working practice of the group was to raise their levels of computer literacy. The project introduced a network of desktop computers in advance of the organisation's move towards increased computer utilisation for office based tasks. The impact of the organisation's disposal of typing pool facilities was reduced by the previous spread of computers to the desk of each individual group member. The installation of an internal network, allowing group members to transfer and print documents, permitted the group to produce reports internally before this became an organisational requirement.

10.5 Developer Time Allocation

10.5.1 Organisational Record

A time allocation log was kept as an organisational requirement by all members of the organisation. The time log data was reported earlier, (section 4.3.6), as part of the original project specification process. The time allocation from the organisation's staff was based on a weekly reporting structure. The conditioned hours of each individual totalled 37 for each week. With the exception of periods of arranged overtime, hours worked above or below this total were not recorded.

The broad categorisation applied by the organisational data log made it necessary for the developer to further define the categorisation of effort within a separate log. This log was completed on a weekly basis with additional notes kept during development.

The developer's log of time allocation applied clear criteria for the categorisation of effort. The interdependent nature of in-house development and the regular work
programme reported in the previous chapters made categorisation of effort difficult. In order to separate the effort attributed to a particular DAVE project from other tasks it was necessary to set criteria for definition. These criteria were not fully defined at the start of the project as at that point many of the features of the development project were yet to emerge, such as iterative testing and computer support. Table 10.1 presents a list outlining the developer's criteria for categorisation when completing the time allocation log.

<table>
<thead>
<tr>
<th>Category</th>
<th>Included Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Specific</td>
<td>Code generation/iteration</td>
</tr>
<tr>
<td></td>
<td>Graphics generation/iteration</td>
</tr>
<tr>
<td></td>
<td>Database investigation/acquisition/confirmation</td>
</tr>
<tr>
<td></td>
<td>User feedback/iterative testing</td>
</tr>
<tr>
<td></td>
<td>Formal evaluation</td>
</tr>
<tr>
<td></td>
<td>In-house software support and maintenance</td>
</tr>
<tr>
<td>Other</td>
<td>Work programme specific tasks</td>
</tr>
<tr>
<td></td>
<td>Commercial software maintenance</td>
</tr>
<tr>
<td></td>
<td>Hardware maintenance</td>
</tr>
<tr>
<td></td>
<td>Miscellaneous software utilities</td>
</tr>
<tr>
<td>Excluded</td>
<td>Leave of absence</td>
</tr>
</tbody>
</table>

Table 10.1 Time Allocation Categorisation Criteria

10.5.2 Limitations of the Time Allocation Data

The developer's time allocation record offers a means of considering the influences on the progress of the development project and provides an objective measure of the cost associated with development. However, consideration of these issues must take into account the limitations of the data. The limitations are considered to be:

- Inaccuracy related to weekly reporting.

As reported earlier, the weekly completion of the log may result in a low level of accuracy preventing the confident comparison of effort on a weekly basis. A daily record
would have allowed development effort to be considered at a finer level. However, it is considered that the applied approach provided sufficient detail to consider the balance of tasks and to base a judgement of overall product cost.

- The absence of a measure of user effort.

Whilst reporting developer effort, the results do not account for user involvement. Section 5.5.6 recommended informal user involvement to overcome the problems associated with formal allocation of user time. This became a central part of the prototype development cycle for module design. Maintaining a record of informal feedback was considered inappropriate, preventing the study from quantifying the level of user effort beyond the formal processes. The informal approach to feedback was effective from an organisational perspective by shifting user effort from the core time of their work programme to user break periods. As feedback was voluntary this did not become a source of concern to the user group. However, adopting this approach for future development projects demands a careful assessment of user group opinion as withdrawal of user feedback would remove the development cycle's prime means of design tailoring.

- Inability to account for growing experience.

The results show the time allocated by the developer as opposed to developer efficiency. The time spent developing the first DAVE system module included frequent references to the development tool's language manual and numerous iterations of basic code routines as more efficient approaches were revealed. Consequently, although developer effort over the period of the project could be considered approximately constant for each allocated hour, the results of the allocated time were improved as experience grew. This was also noted for user feedback which was found to be more applicable to the development process following experience of past projects, (section 7.3.2).

- The limitations imposed by categorisation.

The effort allocated to specific tasks cannot be completely separated from the developer's other tasks throughout the project. Development within an in-house environment brought with it a mix of influences between the project's many tasks.
Whilst participation in the group's regular work programme offset direct project activities such as developing module code, the insight it provided into the user's requirements influenced the subsequent approach to design, the gathering of feedback, concept suggestion, etc. Similarly, the effort applied to one development project contributed to subsequent projects through the donation of elements, definition of standards, solution of design problems, etc.

10.5.3 Time Allocation Results

Figure 10.3 presents the results of the developer's time allocation log for the full period of the DAVE project. The results are shown as a percentage of total allocated time for each month, (figure 10.3). They provide a view of the trends in development effort and the relative balance of effort between projects.

At the start of the project, time allocated to 'other' tasks dominates the table. This relates to the intended induction period. The developer was included in the group's work programme, assisting other members of the group in tasks including ergonomic assessment, field trial support and report preparation. An example is provided by the peak for October/November 1991 during which the developer was a co-author for two APRE reports.

This balance changes with the start of the DAVE project's initial specification process. This is joined by the investigation of software development approaches with the construction of the test applications. The Numerical application met its aims of functional testing for the lowest relative time cost. The more costly Text and Interface applications were developed beyond their initial goals. One view of this extended effort is that it represents a lack of focus on the DAVE project's goals. However, the subsequent value of these applications as a models for future development and as an extension of the group's capabilities, suggests that the loosely controlled software design exploration benefited the user group. Whilst apparently effective at this point in the project, it cannot be recommended as a development approach where a clear concept of requirements is available. The subsequent DAVE module development projects followed a more controlled process based on a more complete view of the group's requirements and the scope of the development tools.
Figure 10.3 Developer Time Allocation

<table>
<thead>
<tr>
<th>Task</th>
<th>1st Month</th>
<th>2nd Month</th>
<th>3rd Month</th>
<th>4th Month</th>
<th>5th Month</th>
<th>6th Month</th>
<th>7th Month</th>
<th>8th Month</th>
<th>9th Month</th>
<th>10th Month</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bug Fixes</td>
<td>4.11%</td>
<td>1.02%</td>
<td>4.87%</td>
<td>4.05%</td>
<td>3.72%</td>
<td>5.44%</td>
<td>2.61%</td>
<td>6.49%</td>
<td>5.21%</td>
<td>0%</td>
</tr>
<tr>
<td>Other</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specification Test</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Maintenance</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Testing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Analysis</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Design</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Documentation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Content</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Marketing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sales</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Customer Support</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>General Support</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Other</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>39.40%</td>
<td>26.6%</td>
<td>11.83%</td>
<td>9.22%</td>
<td>4.60%</td>
<td>6.49%</td>
<td>2.61%</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: This figure shows the allocation of developer time across various tasks over a 10-month period.
Completion of the DAVE system specification phase marks a shift in developer activity towards project specific activities. At the start of 1992 developer effort was directed mainly towards the development of the DAVE system structure, the 'entry screen' and its first module. Although occasionally interrupted by work programme specific tasks, design of the Anthropometry module became the main focus of the developer's work programme.

Completion of the Anthropometry module was followed by concentration on the design and data set acquisition of the Standards module. Despite its relative complexity, this module was able to draw on code and design solutions arrived at through the development of the preceding applications. As a consequence, the associated time allocation was comparable with the first module. Development of the Standards module shows an initial period of concerted activity in which its graphics and functional code was developed. This is followed by a prolonged period of activity at a lower level to supplement the module’s data set.

The development of the Standards module was required to share effort with the main supplementary application, the Map Simulation. Effort applied to this additional project increased towards the delivery deadline defined by the Map field trial. This shift in effort was made possible due to the group's complete control over the development project, as the single customer for the in-house software. Their designation of the Map Simulation as a priority allowed them to accept the knock-on effect of the delay in the Standards module project.

Development of the Vision module demanded novel design solutions raising the required effort to arrive at an acceptable solution. Despite this rise, the total effort allocated by the developer was lower than reported for the previous modules. This was due to the balance of effort with the second group member who was able to share the development tasks. Although the other group member did not formally record the time allocated to the project the project required a similar level of effort from both individuals effectively doubling the time cost reported in figure 10.3. When considered in light of the eventual absence of requirement for the vision assessment technique, retrospective justification of the project becomes difficult.
The evolution of the Standards module into the General and Questionnaire modules followed completion for the Vision module. The reduced development effort reported in for these modules, (chapter 9), was reflected in the comparatively low developer time allocation. As noted for the Standards module, their development involved an initial period of concentrated design followed by the prolonged activity of data set acquisition. Notably, subsequent maintenance of the General module in terms of updating the module’s database exceeded the time allocated to its original development. The evolutionary development of these modules was joined by a return of the balance of developer activities towards ‘other’ tasks. This coincides with the organisation’s transition to the tri-service Centre for Human Sciences, (section 2.7). One consequence of this change was to place different demands on all members of the group, including the developer.

10.5.4 Product Cost

Whilst taking the reported limitations into account, the record of developer time allocation provides an objective measure from which the approximate cost of each in-house product can be determined. This offers a measure which, in combination with an assessment of benefit based on product usage, provides a basis for the subjective judgement of product success.

Table 10.2 presents the recorded totals of developer effort for the development and maintenance of the reported products. Development hours include time allocated before delivery. Maintenance hours include all remaining time allocation after delivery, including upgrades. The results are presented as allocated hours. A monetary cost is not attached to this value as it is relative to the management arrangements of the project removing any relevance for future development projects.

Additional costs include the computer hardware purchased to support the development and delivery of the product. The initial cost of the hardware and development tools was allocated solely to the development project. However, the subsequent spread of computers within the group was justified by other organisational demands. In relation to the cost of developer effort, the hardware and software represents a minor cost. It should be recognised that development within the current organisation would require
only the purchase of appropriate development tools, due to the widespread implementation of the desktop computer.

<table>
<thead>
<tr>
<th>Project</th>
<th>Development Hrs.</th>
<th>Maintenance Hrs.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text application</td>
<td>193</td>
<td>not delivered</td>
</tr>
<tr>
<td>Numerical application</td>
<td>86</td>
<td>not delivered</td>
</tr>
<tr>
<td>Interface application</td>
<td>189</td>
<td>181</td>
</tr>
<tr>
<td>Anthropometry module</td>
<td>723</td>
<td>270</td>
</tr>
<tr>
<td>Standards module</td>
<td>654</td>
<td>120</td>
</tr>
<tr>
<td>Vision module</td>
<td>308 *</td>
<td>101</td>
</tr>
<tr>
<td>General module</td>
<td>177</td>
<td>209</td>
</tr>
<tr>
<td>Questionnaire module</td>
<td>189</td>
<td>123</td>
</tr>
<tr>
<td>Lab Tools</td>
<td>127</td>
<td>92</td>
</tr>
<tr>
<td>Map Simulation</td>
<td>438</td>
<td>107</td>
</tr>
</tbody>
</table>

* Excluding the effort of the second developer

Table 10.2 Project Time Allocation Summary

As noted earlier, the categorisation of effort fails to take account of the project's contribution of work to later projects and other areas of the work programme. A more holistic view might be appropriate. Taking this approach, the product of the development period could be considered to include the range of benefits in addition to the specified modules of the system. The delivery of supplementary applications, the expansion of group capabilities, the achievement of regular work programme goals, the growth in the group's computer literacy, etc. might all be judged against a single measure of the developer's total time cost. Whichever view is taken, the cost/benefit analysis within this specific application niche remains a subjective judgement of value which rests finally with the customer. In the case of the DAVE system, continual assessment of the project's benefit led to a renewal of the contract in October 1993 indicating an positive judgement of the project's value. However, this conclusion was dependant on potentially variable criteria held by the organisation and the individuals involved in the project.
10.6 Discussion of Methods

The results presented in this chapter provide an objective measure as a basis for comparing the various in-house software products delivered over the course of the DAVE project. These findings, in conjunction with the results of formal usability evaluations, allow conclusions to be drawn concerning the overall acceptability of each product. These results have been used in conjunction with the associated costs to assess their relative success. As each product was developed following a variation on the project's evolutionary lifecycle, the success of each guiding model and its components can be reviewed by association with the success of the product.

10.6.1 Project Selection and Module Specification

The preceding chapters combined to demonstrate the ability of the 'modular system' lifecycle's approach of user centred product evolution to determine and meet the specific needs of the target population. Formal evaluation of each product showed that they provided the specified functionality in an accessible form for all members of the user group. Whilst these results suggest that the design objectives of each development project was successful, the usage results show a range of product success in terms of actual usage. The minor impact of some modules on the group's work programme questions whether project selection and module specification was appropriate in these cases.

The overriding factor determining product success appears to be the work programme's demand for the information offered by the tool, rather than the ability of the product to meet its defined goals. Despite providing an effective tool to gather and guide an improved technique for vision plot collection, the Vision module can be considered to have failed to support the work of the group. The cause of this failure was not considered to relate to the design of the product as it was shown to be both usable and functionally complete at the time of formal user testing. Instead, the module was neglected due to an absence of demand from the group's work programme.
When balanced against its cumulative cost of development, the Vision module project must be considered to be unsuccessful. Development effort might have been more suitably applied towards an alternative development or maintenance project. This conclusion depends on the knowledge gained after development. The issue confronting the project's approach to development is whether the project selection process or specification task should have rejected the Vision module concept. To be effective, the project selection process should have highlighted the risk of an absence of work programme requirement for this tool and balanced this risk against the potential benefits of the product.

Whilst recognising the risk of low usage was recognised by the joint specification processes of both the HUFIT tools and the 'modular system' lifecycle's specification phase, a complete absence of requirement was not predicted by the group on whom these processes relied. The group's perception of the high value of a new evaluation method overruled concerns of infrequent use.

If used at the predicted levels, the Vision module could have been considered successful. Starting at such a low expected usage rate amplified the consequences of the eventual fluctuations in the work programme. A few extra vision assessment projects would have exceeded the expectations for the software. In the event, a similar fluctuation led to its neglect. The long period between delivery and an opportunity to apply the software removed the impetus for use provided by the product's launch. By the time an opportunity arrived to use the module, a lack of personal confidence in its application contributed to its rejection for field based use. If the group's work programme had continued to demand vision assessment at the same frequency required before module development, concerns about the approach to project selection would not have been raised.

In contrast to the Vision module, the other delivered in-house software displayed varying but continuous levels of voluntary usage. Whether these levels of use constitute acceptable value for the applied development effort remains a customer judgement. The judgement was effectively made firstly by the groups renewal of the developers contract to continue the project and secondly in their acceptance of the project's final report, (Beagley 1995), for the conclusion of the project.
10.6.2 Evolutionary Design and Development

Complete rejection of a product would have raised concerns surrounding its design and development. However, each module achieved a consistent level of use which, although lower than expected for some modules, confirmed that its level of functionality and usability met the core requirements of the users. Confidence in the acceptability of the functional designs was consistently supported by the results of formal user testing for each module. However, the failure of some modules to achieve their anticipated levels of use or to replace the alternative information sources raises questions about their design or the basis for setting product expectations.

The Anthropometry module was developed to serve the group’s highest priority information area, as specified at the start of the project. The specification process was unable to make a reliable estimate of the likely frequency of enquiries for this or any other specific information area. As the product was seen to provide the group with a new information resource, its level of use was a lower concern than the accessibility and accuracy of the data. The stabilised level of use was considered to reflect the actual level of anthropometric enquiry carried out by the group. The cumulative cost of its development and subsequent design evolution was the highest of the reported modules. As the first module to be delivered it was unable to draw on experience and a library of components. Although the level of effort was relatively high, it reflects the time period for which it was available to the group. In addition, much of the project’s development experience was gained through the development of this module.

The level of Standards module use was consistently higher than the other modules reflecting its wider application to the work of the group. In contrast, the General module did not achieve the expected usage levels due to the failure of its document database to expand beyond the range of reference documents supplied pre-indexed by the developer. Whilst the group made use of these miscellaneous documents they lacked either the requirement, motivation or confidence to acquire their own documents. A alternative strategy might have been to allocate the task of all document acquisition to the developer. This could have been undertaken as an ongoing maintenance activity leading to the continual expansion of the module’s database and consequently its applicability to the
group. The module continued to achieve a consistent level of use despite the project's failure to actively promote the facility.

The usability of the General module and Questionnaire modules had been confirmed by the formal evaluation of the Standards module. Consequently, the usage levels are considered to indicate the actual work programme enquiry levels for each information area. The relative development costs of these modules are reflected in their subsequent levels of usage. Similarly, the relatively high cost of Standards module development was in line with its dominance amongst the in-house products.

10.6.3 Maintenance and Computer Support

The transfer of code and graphics between modules and the continual rise in experience of both the group and the developer blurs the line between development and maintenance within the DAVE system project. The process started with the design of the test applications. Some aspects of the Anthropometry module's design relied on the evolution of concepts and components implemented within the test applications. Subsequently, the Standards module adopted design solutions for text searching and retrieval reached through the exploration of preceding projects. In the cases of the General and Questionnaire modules, new design features accounted for a small proportion of the final products which were able to evolve from an existing product. As the project progressed, development became more efficient. A library of functional components was built up over the course of the project avoiding the need to address many of the smaller design problems for which solutions had already been developed.

The justification for the maintenance of delivered modules requires an assessment of benefit against the associated cost. By shortening the development cycle, based on a reliance on past evaluations for design guidance and informal feedback for validation, the maintenance cycle required less effort from both the developer and the users. The time allocation data confirms that the evolution of the General and Questionnaire modules required less development effort than the equivalent Standards module. However, this was balanced by their lower levels of use. This is not to suggest that there is a link between development effort and eventual usage. Had the Standards module been developed as an evolution of the Questionnaire module, its use could be expected to
achieve an equivalent level despite a lower relative development effort. However, it is probable that the total development cost for this group of applications would have been similar irrespective of their order of development. What the usage results show is that the specification process correctly identified the information areas of highest demand. The benefit of the reduced development effort associated with continued evolution was the project's ability to expand the functionality of the DAVE system at an appropriate level of added effort for the perceived benefit of the new tools.

Computer support is illustrated in figure 10.3 by occasional periods of effort attributed to specific products after the module's delivery. This does not provide a definitive measure of the computer support requirement. Many of the computer support tasks do not appear in the time allocation data as they were too brief to appear in the scale of hours per week. However, it should be noted that the 'other' category of tasks includes work relating to hardware and commercial software support where a greater demand for support was placed on the developer and by association, the project's resources.

The project's characteristic of close association between the developer and the user group was considered to reduce the risk of product neglect. By being present on-site, the developer was able to guide the group, encourage use of the software and correct software problems as they arose. In comparison, many commercial applications were only used for a brief period followed by neglect and eventual removal from the group's computers.

10.7 Conclusions

The application logs and time allocation reports have provided a basis for a cost/benefit analysis of the delivered products and the lifecycle processes responsible for their design. With the exception of the Vision module, the delivered in-house products were considered to have achieved success in terms of consistent usage. Whilst the justification of the development cost remains a value judgement, in this case the project was considered by the customer to be successful on the grounds of its extension of group capabilities.
The conclusions drawn from the consideration of the cost/benefit analysis of the in-house software products reported in this study are presented below.

1. Usage levels of information system modules commonly peaked following delivery due to product exploration by the user group.
2. The usage levels of the delivered DAVE system modules declined gradually over time.
3. Upgrade of the Anthropometry module prompted a temporary increase in its use.
4. The breadth of information offered by the Standards module made it widely applicable to the group's work programme leading to its relatively high levels of use.
5. Voluntary use of applications following delivery provided an unbiased measure of each product's value to the group.
6. Development of supplementary applications, beyond the initial scope of the project positively contributed to the work programme of the user group.
7. Developer presence reduced the risk of the group neglecting occasionally required software by providing maintenance and instruction on call.
8. Document searches using the DAVE modules were used in conjunction with the paper source which was preferred for reading once the reference had been located.
9. User demand for a product at the specification stage did not guarantee its use despite achieving acceptable standards of usability. The users needed experience of the product to be certain of their requirements.
10. Products designed for occasional use were sensitive to fluctuations in the group's work programme.
11. Continued evolution through the reduced maintenance cycle allowed the functionality of the DAVE system to be expanded without the relatively high development cost associated with design from first principles.
11 A 'Modular System' Lifecycle

11.1 Chapter Overview

Chapter eleven draws together the study's conclusions relating to small scale, in-house software development. These conclusions are based on the observations and formal evaluations of the preceding case studies in addition to an analysis of each product's benefits in relation to their associated costs. The findings of the study are incorporated in a guiding methodology for development. A framework model of this 'modular system' lifecycle is presented, followed by process models of its component phases. A combined model is provided as a guide to future development projects.

The chapter starts with a review of the study's background, highlighting the complementary aims of the project and the linked study. The project lifecycle's evolution from a sequential model to its final cyclic structure is traced. The structure is discussed as an approach for managing the characteristics of the development project and its environment. The subsequent sections discuss the component processes of the 'modular system' lifecycle. Each section presents a recommended process model to meet the requirements of the separate lifecycle phases. The conclusions leading to their design are reviewed and the limitations of each design are discussed.

The 'system level' specification phase is presented as the hub of the lifecycle model, responsible for directing either development or maintenance. The changing role of this phase over the course of the project is discussed. The separate development and maintenance cycles are then considered at a 'module level'. The function of detailed specification within the alternative development and maintenance phases is examined. A common feature of each cycle was the approach of design exploration through user centred prototype iteration. The advantages and drawbacks of the project's prototype development cycle for design are reviewed and a process model is presented as a guide. The project's acceptance phase is presented as a model for product confirmation based on a mix of user involvement and considered formal evaluation techniques. The model concludes with the process of product delivery and a mechanism for continued product support, addressing all phases of the product lifecycle within a changeable environment. The 'modular system' lifecycle's structure and component processes are presented in an
integrated model as a guide for future small scale in-house development. The chapter concludes with a discussion of the study's limitations and recommendations for further work.

11.2 Background

11.2.1 The Scenario

The DAVE project was initiated by a defined user group within a small, research centred organisation. The project differed considerably from the traditional software development task. Rather than realising a customer's advanced concept of a product, the task was to explore the potential for software support of the group's work programme. The approach taken led to sequential delivery of products tailored to the emerging requirements of the users.

The work of the VDS group and its place within the organisational structure of the APRE were described in chapter 2. The organisation's hierarchy and the specialist disciplines of its experts led to a divisional arrangement in which separate groups supported identifiable scientific niches. Each group met their work programme commitments through internal resources, retaining control over their own budgets and work programmes. The DAVE project was funded from the VDS group's resources providing them with full control over the project and its delivered products. Whilst the project's aim was to "To assist the work of a group of experts in the field of vehicle ergonomics" (Beagley et al. 1992), the means of providing this support was not defined by the customer.

11.2.2 The Study

This study has covered a five year period of the DAVE development project from its inception. This has included the analysis of requirements, selection of tools and all aspects of the software lifecycle from specification, design and development through to the delivery and maintenance of a range of products.
The study and the development project provided mutual support. The study's literature review, (chapter 3), gathered related guidance as a grounding for the development project. The study continued to support the development project with input from current literature throughout the project. A small number of references were found to relate specifically to the implementation of computers within small organisations. Research was found to be concentrated on the development of large systems, e.g. (Royce 1987, Curtis et. al. 1988, Harker 1988, Myers 1990). However, the continuing spread of computers both within the workplace and the home, along with the emergence of simplified software development environments, was considered to bring the potential for software development within reach of the small organisation (Grudin 1992).

The project's analysis of requirements led to the selection of in-house development as the preferred method of meeting the group's demand for computer based support. The suggested scope of new 'high level' programming tools, (Nicolson 1990, Young et. al. 1990), raised the possibility of developing an in-house programming capability within the limited resources available to the project. Examples of the use of 'prototyping tools' to deliver novel software applications continued to appear throughout the project's time-scale, e.g. (Coleman et. al. 1993, Poulter et. al. 1993, Beroggi and Wallace 1994).

The pitfalls of software development highlighted ranged from code failure, (Corbato 1991), to system atrophy (McCosh 1984). Factors considered to influence the success or failure of software products centred on the need to account for the specific requirements of the users and the host organisation (Nickerson 1981, Harker and Eason 1992). The DAVE project's scenario of in-house development offered the possibility of addressing these issues through a user centred approach.

The literature review offered a range of guidance from detailed structured methodologies such as SSADM (Edwards et. al. 1989) to lifecycles proposing a process sequence for product development (Royce 1987). Specific guidance for software development for projects matching the characteristics of the DAVE development programme was not found. This provided an opportunity for the parallel study to contribute to this emerging area by tailoring a development approach to guide small scale, in-house software development.
11.2.3 The Development Project

The DAVE project was initiated by a customer's demand for computer based support for their specific work programme. However, the user group did not hold a firm concept of which areas of their work programme should be supported or the form a solution might take. The project sought to investigate their requirements and explore computer based solutions.

The project started with an analysis of the organisation and the group's work programme to determine a strategy for support. This revealed a wide spread of relevant reference sources, unsupported by commercial software at that time. The parallel development of test applications confirmed the suitability of suggested development tools for both prototyping and the delivery of the final product, (chapter 5).

In-house development of tailored software products was accepted by the group as the chosen approach to acquiring computer based support for their work. The project relied on the associated study to select an initial development approach and to refine a lifecycle model to match the particular characteristics of the project as they became apparent.

Over the study time-frame, the DAVE system grew to include five modules covering different user specific information sources. The in-house programming capability was further exploited to produce a range of supplementary applications, (appendix H), expanding the group's repertoire of research techniques. The DAVE development projects provided a series of case studies which implemented and evaluated a tailored lifecycle model leading to its evolution. The resulting methodology provides both broad development guidance through its modular structure and detailed guidance in the form of process models for each phase of activity. The following sections set out these models.

11.2.4 User Participation

The development project was initiated by a small group of users who remained the focus throughout. User involvement was considered to be an essential part of the development process by both the users, who held a stake in the product's success, and the developer, who saw close user contact as the key advantage of the in-house development scenario.
What was unclear was the level and form of user participation necessary to guide acceptable design. The study's 'modular system' lifecycle grew out of the project's attempts to balance user effort with the value of their contribution to the development process. The tools and techniques chosen to deliver the DAVE system sought to maximise the effective input of the user group within the constraints of available effort. This standpoint led to a form of participatory design (Eason 1988, Macaulay 1996) in which user participation and mutual reciprocal learning contributed to all stages of the product's lifecycle.

Kawalek and Leonard (1996) reported a case study in which participatory design extended to the encouragement of the user population to develop the skills necessary to tailor software solutions to their individual requirements. In doing so, they highlighted the need to take an evolutionary approach in order to adapt to environmental changes. Whilst the DAVE project was required to respond to these changes, this level of user participation was found to be both impractical and undesirable. Although user participation remained central to each lifecycle phase, the emphasis rested on the developer acquiring the necessary development skills as well as a practical understanding of the group's work programme. The group remained focused on their work programme, relying on the developer to support and modify the delivered system, (section 4.6.7).
11.3 A ‘Modular System’ Lifecycle

Figure 11.1 The ‘Modular System’ Lifecycle Framework

Figure 11.1 presents the structure of the proposed model for in-house development of a modular system. The following sections describe the origin and evolution of this model.

11.3.1 Model Design and Evolution

The traditional sequential lifecycle, (figure 3.2), was selected as an initial plan for the project's work programme. The large scale formal methods were judged to be
inappropriate for the DAVE project's limited resources. As specification of requirements was a common starting point for the reviewed development models, the simple 'Waterfall' model was considered to provide a suitable model to initially guide the project. Conflicts between this lifecycle's linear view of the project and the actual development task were anticipated. The study aimed to modify the traditional model's basic structure to match the project's development scenario as its features became apparent. The model was revised following the DAVE project's original 'system level' specification phase to match the characteristics of the first development project.

The project's interpretation of the traditional model's specification phase led to the selection of a single information area for immediate development. The group was unable to adequately specify the form and function of the proposed product, matching the observations of Luqi (1989) and Spence (1991). The project responded by using the chosen prototyping tool to explore design solutions for the identified database with the group. Whilst the sequential lifecycle structure was retained, the phases were modified. The revised model, (figure 6.1), relied on the approach of prototype evolution, as defined through test application development, to overcome the users' inability to firmly specify the product.

Evaluation of the first DAVE system module concluded that the evolutionary lifecycle produced a product which met the functional requirements and usability standards required by the customer group. Its approach of prototype evolution from a minimal critical specification was accepted as a viable alternative to detailed specification for the given scenario and followed as a guide to the development of the DAVE project's subsequent modules.

Delivery of this first module was followed by a demand for further development. However, the linear lifecycle used to guide its development placed maintenance as a final perpetual phase in the lifecycle. Had a separate maintenance team been available to support this role, then a new development project could had started following the linear sequence. However, the project was limited to an individual to complete both the tasks of development and maintenance. Support in the form of a modular system demanded that effort should be applied to the development of a new concept product following the delivery of a module. The proposed solution was a 'modular system' lifecycle, (figure...
11.1), in which a global view of the project is taken to choose between the alternatives of the development of a new concept or the maintenance of a delivered module.

The 'modular system' lifecycle separated the development task into several levels of detail. Firstly, specification is presented as a 'system level' task in which the requirements of the users and the organisation are reviewed prior to the group's selection of the project's continuing work programme. This accommodated observed shifts in requirements related to factors such as the widening of technological options, changes in the group's work programme and the users' growing understanding of the project's scope. Concepts selected for development or modules selected for maintenance would then considered in a more detailed 'module level' phase. The combined design and development activity, in the form of the prototype development cycle, breaks product assessment down further to consider design both in terms of the prototype as a whole and the detailed design of its elements, (figure 11.5). The lifecycle returns to a module level view for evaluation and delivery before returning to the start of the lifecycle for an updated global view.

This cyclic lifecycle model was followed with a return to 'system level' specification following the delivery of the DAVE system's first module, (section 7.2.2). Although a priority list had been constructed at the start of the project, (table 4.5), the lifecycle demanded a system level review before the start of a new module development project. This cyclic approach to the control of the project work programme allowed changes in user requirements and project scope to be considered with the benefit of increased experience for all concerned. The result was the selection of a product concept which had not been considered at the previous specification meeting but went on to become the most used module of the DAVE system.

The ability of the 'modular system' lifecycle to support the full demands of the project was tested by the selection of the maintenance cycle, following delivery of the second DAVE system module, (section 9.4). The maintenance cycle was able to build on the techniques successfully applied to module development. The easy access to the code and interface elements of the delivered products allowed their upgrade to be approached as a continuation of the evolutionary cycle used to develop the product. This approach was used to bring the first DAVE module into line with the more advanced presentation approach that had become the standard for the later modules. The cycle was shortened by
using a delivered module as a template and by making assumptions of acceptability based on the module's formal evaluation, carried out as part of its original development cycle. User involvement was retained in the cycle, providing a further control over product suitability. The reduced maintenance cycle was used later to develop new products based on the template of another delivered module. The assumptions of acceptability were upheld by the stable levels of use achieved by the maintained modules.

The original version of the 'modular system' lifecycle model suggested that all maintenance tasks be supported by a formal maintenance cycle, (figure 6.9). However, the close working environment of in-house development was found to place immediate demands for computer support on the developer. In the same way that the users were on-hand to provide design feedback throughout the development and maintenance cycles, the developer was accessible for user enquiries and maintenance requests. The majority of requests for support required a small level of developer effort and could not reasonably wait until a maintenance cycle had been initiated. Correction of small sections of code or design problems were completed immediately, minimising user frustration to ensure continued support for the DAVE system. Similarly, requests for instruction were always met on demand. Although user demand for instruction was infrequent for the DAVE modules, the drift in organisational culture towards the use of computers for all office activities led to an increasing number of requests for assistance in using commercial software and supporting the computer hardware. This expanding role is represented in the final 'modular system' lifecycle model as a 'floating' computer support role. These actions were essential to the efficient function of the group but could not be linked to identifiable points in the project's lifecycle.
11.3.2 Phases of the 'Modular System' Lifecycle

Figure 11.2 presents the phases applied to the cycles of development and maintenance for the DAVE project.

Figure 11.2 Phases of the 'Modular System' Lifecycle
The development cycle sets out a more comprehensive work programme. The loose specification provided by the 'system level' specification is further defined at a 'module level' leading to the design of a prototype. The design evolution towards the requirements of the group is then progressed by developer and user centred iteration within the model's prototype development cycle. A formal acceptance phase provides a control over product quality prior to delivery. The cycle ends with a return to 'system level' specification to consider the system as a whole before specifying the continuing project work programme.

The maintenance cycle follows a reduced model. Two central assumptions are made to reduce the effort associated with the maintenance. The first is that the modification of an existing design requires a simplified re-specification activity as compared to the more detailed specification undertaken for the design of a new product. The second saving assumes that user involvement in module redesign provides an adequate safeguard against the delivery of an inappropriate design changes, whilst previous formal acceptance of the template continues to confirm the acceptability of the core module. Whilst radical design changes might prompt formal evaluation of the final product, any project making fundamental changes to a module's function should be considered as a separate development project and follow the lifecycle's development process. At the centre of the maintenance cycle remains the project's preferred design approach of the prototype development cycle. This approach to maintenance of continued module evolution depended on the easy access of the programming code and interface elements allowed by product design and delivery using a the same high level 'prototyping' tool.

Each of the lifecycle's phases are discussed in more detail in the following sections. A process model is presented for each phase and the conclusions leading to the proposed design are reviewed.
11.4 'System Level' Specification Phase

Figure 11.3 presents a process model for the 'system level' specification phase of the lifecycle. This phase was the hub of the 'modular system' lifecycle, responsible for managing the DAVE project by directing the project's work programme. The role of 'system level' specification changed throughout the project. At the start, it provided the background information used to plan the DAVE project through the activities of requirements analysis and platform/tool selection. Later in the project, the 'system
level' specification phase became the point at which the system was reviewed to update the view of the group's requirements. This overview was used to inform the group discussion process in the selection of the project's continuing work programme.

11.4.1 Background Analysis

At the start of the project the user group, who were also the customer, had an indistinct concept of their requirements beyond their asserted need for computer based support to their specific work programme. It was necessary for the project to help the user's define their requirements and determine a method of supporting them. Three sources were identified as the basis for an analysis of requirements. These were:

1. The group's work programme - Based on direct participation and observation.
2. The members of user group - Based on interview and voluntary log completion
3. Historical data - Through analysis of the organisation's records.

Induction of the developer within the user group allowed the project to draw on the advice of group members at any point throughout the project. Whilst not a formally reported source of information for the specification of the DAVE system, informal discussion played an important role. It allowed the issues raised by the formal methods of user interview and questionnaire completion to be considered and discussed without the restrictions of the organisation's time accounting procedures.

The formal approach of user interview was important for ensuring equal representation of views across the group. By taking a semi-structured interview approach a valuable two-way discussion was prompted. Although time consuming, the interview approach was manageable given the relatively small numbers of users. The voluntary completion of work logs was found to provide only a limited snapshot of the group's work which was further biased by disproportionate completion. Despite these problems this information supplemented the informal discussion process leading to a list of possible areas for support.

The historical data contained by the organisation's time allocation records offered a promising source of information. It separated the group's work into areas which could
be further characterised in terms of the activities involved and the likely reference areas. However, the lack of detail and possible inaccuracies in the data reduced the value of this analysis task. In addition, the data did not show clear trends in the group's work programme against which a development programme could be planned. The predictions by the head of the group concerning likely work areas for the future proved to be equally sensitive to changes in the organisation and in customer demands.

11.4.2 Platform and Tool Selection

Platform and tool selection was conducted in parallel to the analysis of the information requirement, (figure 11.3). If a commercial product had been found to meet the requirements of the group, it would have been purchased as a package with the required computer hardware. If available, this remains the most appropriate source of computer based support for the small organisation in a continually broadening commercial software market. The decision to develop was based on the absence of suitable commercial software and the assessment that new software tools made in-house development feasible. Whilst the results of this study confirm the feasibility of small scale in-house software development, the associated cost of tailored software was considerably higher than more general commercial applications such as word processor software.

Having excluded high cost hardware and software, a computer platform was acquired to support the chosen development tool. The selected tool suited the development project, compromising between the often conflicting requirements of ease of use and scope for application development. Final acceptance of the tool depended on the determination of the functional limitations related to its high level language. The development of test applications proved the suitability of the tool, provided experience for the developer and raised the possibility of using the tool for both prototype development and product delivery.
11.4.3 Project Overview

The 'project overview' phase describes the global assessment of the DAVE project prior to the group's selection of the next development or maintenance project. The close working environment of in-house development led to the regular discussion of ideas relating to the current and future design of the DAVE system. Project overview offered a defined point in the cycle when feedback was actively sought from the group members. Ideas raised or restated by this process provided the agenda for discussion during the specification meeting.

A criticism of the DAVE project's approach to 'system level' specification was the absence of formal reporting of the overview process. Although the delivered products were reviewed and the user consulted at the end of each development project, these actions were not formally recorded. A more appropriate practice would have been the preparation of a brief review report to mark this stage in the lifecycle.

Another weakness of this phase was the absence of a formal accounting process for comparing development effort against product benefit. Whilst all but one of the modules of the DAVE system were shown to be not only usable but also used, (chapter 10), the decision to proceed was largely subjective. A formal method of judging their potential contribution to the work programme might ensure consistency through criteria based selection. However, this predicted contribution cannot be based solely on anticipated usage but must take account possible changes in the group's capabilities, efficiency and the impact on the quality of work. These were issues which the study was unable to adequately measure, let alone predict prior to development of a concept. Another source of project specific information, unused at this point was the background log data used in chapter 10 to determine each module's levels of use following delivery. This information would have provided a measure of value against which the anticipated cost of subsequent development projects could be justified or rejected. This issue is discussed further in section 11.7.4. Despite failing to use some of the information sources available for assessing the project, this guiding phase succeeded in specifying the information areas which became the basis for DAVE system modules, considered to be successful by the study's measures of usability and usage.
11.4.4 Work Programme Specification

As guided by the 'modular system' lifecycle, the future of the DAVE system's development programme was decided at formal group meetings based on a system level overview of the project. The first formal group meeting established a prioritised list of possible information areas for support. This represented their minimal critical specification for the DAVE system at that time. By gathering further details relating to available data the developer was able to prompt a more detailed specification of the module's requirements. Prototype development based on the resulting loose definition provided the starting point from which the prototype development cycle was able to define the product through iteration, (chapter 6).

Completion of the first module led to a return to 'system level' specification. The approach of group discussion was retained to ensure that the DAVE programme remained accountable to the users. The system overview process had gathered user and developer views, formed from experience of the first module, to raise text retrieval as an addition to the previously compiled priority list. The second formal specification meeting of the DAVE project supported this new concept over the previously suggested products.

It is notable that as the project progressed, the raised experience of the group allowed the formal meeting's discussion to extend beyond proposed information content to address the possible functionality of concept products. The first formal group discussion was limited by the absence of examples on which the group were able to base their concept of the system. The form and functionality of the first delivered module offered the group a frame of reference for the discussion of subsequent module concepts. In this respect the group's knowledge, in terms of experience and appropriate models, dictated the extent of their specification. This added depth of consideration within the 'system level' specification did not remove the need to further define the concept at a module level. Instead it provided a more advanced initial concept against which the details of the available data and group's current practice could be considered.
11.5 'Module Level' Specification Phase

Figure 11.4 presents a process model for the development cycle's 'module level' specification phase. The role of this phase was to provide sufficient detail to support the design of an initial prototype. The equivalent re-specification phase of the maintenance cycle was reduced to a single, user centred, specification task. This was used to define necessary changes to an existing design rather than guiding a novel design.

![Diagram of 'Module Level' Specification Phase]

Figure 11.4 'Module Level' Specification Phase

11.5.1 Module Specification

As noted above, the 'system level' specification phase produced a minimal definition of the product to be developed which identified the area for support. The purpose of
'module level' specification was to tie available information into the initial design of a prototype. From this point, the DAVE system modules depended on design exploration through iterative prototype development.

The tasks of requirement definition, data set selection and function list construction were introduced for the 'module level' specification of the Anthropometry module to get the users to consider the data available and the way in which it could be used. Data set selection required the users to consider the range of information required and the form in which it should be provided. This tied in with the requirement definition process which prompted the user to consider the way they used the target information as a means of specifying the necessary functions of the module. Figure 11.4 presents these tasks as parallel activities. The influence of each on the eventual design depended on the characteristics of the proposed product. As the first two modules were information retrieval tools, their 'module level' specification focused on the extent and form of available data. In contrast, the third module was designed to gather data leading to a concentration on the functions indicated by the requirements definition. The final activity of function list construction was used as an aid to the initial design of the prototype.

The examples of DAVE module development show a change in the form of user input to the 'module level' specification phase over the course of the project. As noted for 'system level' specification, the experience of the group members improved their understanding of the scope of the project and the specification/design process. Specification of the Anthropometry module required the developer to lead the subjects through the consideration of their requirements. The requirement definition took the form of model questions which all members of the group were able to relate to their work programme. Specification of subsequent modules reveals the user's raised understanding of the potential of software systems. User views extended from a consideration of their current approaches to the use of information, to include ways of enhancing information handling.

Although the need to define the group's requirements and establish the module's proposed data set remained the same for each project, the role of function list construction was reduced. Later development projects were able to draw on existing in-
house applications as both models for design and a source of code and graphics reducing the role of the function list as a starting point for prototype design.

11.5.2 Module Re-Specification

Re-specification became a lifecycle process with the introduction of the maintenance cycle. Modification of the Anthropometry module followed a shortened development cycle in which changes to the module’s functionality and interface design were made, (section 9.4). This maintenance task responded to a ‘system level’ specification decision to upgrade the module in line with the DAVE system’s new standard format for presentation across a network, (section 7.5.3).

Re-specification as a ‘module level’ phase enforced a formal review of user requirements before design modification. For all DAVE maintenance projects, the users provided full considered feedback concerning all aspects of the module’s design through individual interviews. The role of the maintenance phase’s re-specification process was different from the equivalent module level specification phase of the development cycle. In place of the requirement to fully explore all aspects of the design was simply a need to assess an existing design for possible improvements. This application model acted as an accessible description of the current specification from which the users were able to determine and communicate necessary changes.

11.5.3 HUFIT Tools for Specification

With the exception of prototype B of the Vision module, the DAVE development projects depended on a cycle of iterative prototyping from a minimal specification which was proposed at a ‘system level’ and refined at a module level.

The Vision module took two competing approaches to design before comparing the prototypes and delivering a combined product. This allowed the DAVE project’s approach to specification to be contrasted with the more detailed approach set out by the HUFIT toolset. The designer of prototype B, following the HUFIT toolset, was left with the task of producing a design which did not conflict with any of the specified requirements. The
design of prototype A approached from the other direction by producing a design based on a minimal specification. This design was then iterated to overcome problems as they were revealed by user interaction. Following this approach, the prototypes of the DAVE system modules acted as their own specification documents.

The HUFIT tools guided the designer in the consideration of the issues surrounding product design such as user characteristics, the task environment and the intended use of the product culminating in the completion of functionality matrix. The matrix was applied to cross reference user and task requirements against the proposed functionality of the product. The HUFIT tools leading up to this point provided a suitable structure to the specification process. However the matrix was less applicable to the development scenario as it relied on a list of specified functions, which the users were unable to adequately provide.

The eventual Vision module drew predominantly from the evolved design of prototype A rather than the HUFIT specified design of prototype B. The 'modular system' lifecycle's approach of design evolution advanced the combined prototype to the point of delivery. It was concluded that for the given development scenario the emphasis should be placed on design evolution rather than detailed specification. However, this does not exclude the use of tools, checklists and guidelines as a source of guidance and a means of ensuring consistency between development projects when more than one developer is involved.
11.6 The Prototyping Phase

Figure 11.5 The Prototyping Phase

Figure 11.5 presents a process model for the prototype development cycle. This cycle provided a focus for the development and maintenance of all the in-house products reviewed.

11.6.1 The Prototype Development Cycle

The software development tool was chosen for its balance between ease of use, (Nielsen et al. 1991), and the scope for its products, (section 5.5.3). Whilst SuperCard applications executed more slowly than lower level languages, the development tool supported simple construction and modification of fully functional applications. Test application development revealed ways of avoiding unacceptable feedback or processing.
delays. Their development demonstrated the tool's ease of use and the functional scope of its products, confirming the possibility of product development within the project's limited resources.

In addition to the issue of resource constraints, the DAVE project was faced with the absence of a clear customer concept. The detail in the customers' stated requirements was constrained by the information available on which they could base their concept. Initially, their relevant experience was limited to their working domain. Consequently, their confidence in specification ended with the selection of a topic for support. The group was not in a position to form or express a common product concept without a software example as a frame of reference. The development tool's ease of use and flexibility bridged this gap by allowing creative design solutions to the group's minimal specification to be prototyped and presented for review. In doing so it provided the information required for the group to consider specification at the next level of detail, product design. Over the course of prototype evolution, further design issues would be raised and solutions considered through the accessible medium of the interactive prototype.

The study sought to tailor this process to the development scenario. The result was the design of the project's iterative 'prototype development cycle', (figure 11.5), which became the central guide to all subsequent in-house development. The cycle starts with the outline of a design solution in the form of a non-functional interface. These simple designs helped to express the form and function of design concepts for consideration by the developer and the user group. This allowed the group to arrive at a common concept based on one of the prototyped solutions. Design alternatives could be assessed before committing major development effort. Acceptance of an overall design was followed by the sequential selection and development of prototype elements. Adding functionality to a prototype element often held implications for the overall design, e.g. placing a large document in the 'Text' test application's document window revealed the need to scroll. The prototype development cycle regularly returns the process to a decision point at the centre of the cycle allowing changes to be made throughout the design cycle.
11.6.2 User Input to the Development Cycle

The prototype development cycle provides a process model where the overall design of a prototype is proposed and modified before moving on to develop the suggested components. Following its user centred design principles, the project sought to take advantage of user group accessibility by involving them in this design cycle.

Development of each product within the group's offices provided the users with regular contact with the emerging design. This allowed all group members to provide feedback concerning the current form of the design. Whilst arranging formal assessment sessions would have helped to ensure even control across the group, this approach would have placed an added burden on the users. The informal approach led to frequent feedback from all group members throughout the prototype's evolution in terms of design and functionality.

The prototype development cycle presents two levels of design. The first was responsible for the overall design of the prototype. Cycle A considers the functionality and usability of the prototype as a whole leading to changes in its functional and interface components or even redesign of the overall concept. At a lower level the model considers the design of the prototype's elements. Cycle B involves the selection and design of the prototype's elements. Cycle C represents a further iterative cycle, at this element level, in which the design of the chosen element is reviewed in relation to its specific function before returning to consider its impact on the prototype's design as a whole. The form of the final product hinges on the judgements made at the point of prototype level evaluation, the hub of cycles A and B. This point of evaluation held three alternatives. The first was the redesign of the overall module by either re-specifying its elements or simply rearranging those present. If the overall design was acceptable then evaluation led to the selection of elements for completion or modification. When both the overall design and its elements were acceptable, the design was passed to the lifecycle's acceptance phase.

Although the users were not working to the fixed process model of the developer, their feedback was always at this point of prototype evaluation rather than element evaluation. As noted for user specification, their level of input was related to their
knowledge base. The information provided by prototype designs provide the group with the basis for commenting on design. However, the prototypes did not reveal the code and graphic design issues involved in reaching the promised design. The users did not have the experience to suggest solutions at the level of element design. Whilst they could have been given this experience, that level of involvement in design was considered to be an inappropriate diversion from the group’s main work programme.

The type of design modifications prompted by the prototype, changed as it evolved. User and developer review of the initial non-functional prototype revealed critical features such as design omissions. As its functionality increased more subtle design issues were revealed by user interaction such as apparent confusion related to non-intuitive features. As noted in the review of the Anthropometry module, (section 6.7.4), the potential existed to continue making fine adjustments to the design beyond what was necessary for the product to be acceptable to the users. The project’s principle of accelerated design starts to address this problem with the use of the final acceptance phase to both catch and correct design problems. However, the permutations of creative design prevented the study from attempting to set clear criteria for completion, leaving it instead to developer discretion.

11.6.3 Expert Evaluation within the Development Cycle

Development of the second module extended evaluation within the prototype development cycle by introducing expert evaluation. Expert evaluation had been applied to the lifecycle’s acceptance phase for the development of the first module. Its transfer to the earlier development phase attempted to raise the value of expert evaluation to the eventual design of the module. The minor contribution of expert evaluation to the design of the first module was attributed to its over-development, (section 6.7.3.1).

Expert evaluation was carried out for an early functional prototype of the second module, (section 7.8.3). The recommendations provided were more influential, although less specific than those provided prior to delivery of the first module. The evaluation of an early prototype allowed broad criticism concerning complexity and interface consistency to be addressed in the evolving design. The generalised issues
raised suited the prototype development cycle by providing guidance without constraining the design process.

Expert evaluation was returned to the acceptance phase for the third module in an expanded form, (section 8.10.5). The use of a panel of three experts and the guidance of design heuristics were found to benefit the evaluation process. Whilst increasing the number of experts improved the effectiveness of the evaluation, this was made possible by unusual circumstances which could not be repeated for the project's other modules. Although desirable, it is not considered to be a prerequisite for the success of small scale software products. However, the use of design heuristics as a guide to expert evaluation is a feature which can be readily included in future expert evaluations. The opportunity for an expert to influence the design early in the development process is considered to most appropriately place expert evaluation as a prototype development cycle activity.

11.6.4 Object Oriented Programming

The development approach built on the development tool's ability to encapsulate code in interface objects or standalone routines. The project established a standard coding approach in which all functions and interface components were developed as 'objects'. Examples of encapsulated code routines are presented in Appendix I. The use of 'objects' as application building blocks led to the accumulation of an extensive library of design solutions. The flexibility of the development tool allowed these elements to be taken directly from both delivered applications and smaller utilities.

This provided the development process with an important source of interface objects and functional code. Each 'object' was designed to function irrespective of their location in the interface or the code structure. Graphic elements could be moved and re-sized within the interface. Code elements referred to common variables allowing them to be used in any of the in-house software products. As the project progressed and the library of elements grew, this became an important feature in improving the efficiency of the group's in-house software development capability.
11.6.5 Design Issues

The DAVE system was designed to provide computer based support to a known group of users whose past use of computers was broadly low and isolated to two of its members. The design of the system responded by adopting a 'real world' metaphor for interaction in which the users were presented with a simulated hardware panel, (figure 5.7). User response to this approach led to its adoption as a standard for subsequent DAVE system module interface designs. The 'textured metal' appearance of the original entry screen was adopted as a design standard, indicating each application's identity as a DAVE system module. This approach ensured that the modules remained part of a coherent system following their transfer to separate applications, (section 9.3.3).

By indicating functionality through the impression of raised, interactive 'buttons' and recessed displays, the user group were quick to grasp the full functionality of the interface despite limited computer experience. However, with this approach came the risk of designing to the lowest common denominator. Although providing an appropriate metaphor for the defined user group, literal adherence to the physical model of a hardware panel would have unreasonably restricted the design of future modules. As the functional complexity of the DAVE system modules increased throughout the project it became necessary to apply interface features outside this physical model. Early designs of the Standards module used scroll buttons but were quickly replaced by scroll bars which offered enhanced control over the view of a window. It is notable that the less experienced users initial failed to grasp the full functionality of the scroll bars, simply clicking on the direction arrows rather than dragging the slider or clicking the bar to scroll by page.

It was important that the design approach balanced usability with functionality. The design of the Standards module, (figure 7.3), shows the project's attempt to allow interaction at several levels. Users uncomfortable with the module's search features could simply read the on-line documents page by page whilst more confident individuals could use the module's index list. Finally advanced searching was offered in the form of Boolean filtering. Whilst the integral filtering functions offered the module's key advantage over paper, it was considered to be important to avoid overwhelming the novice user without constraining the more experienced user. This approach allowed the
developer to gradually instruct the novice users in the application of the Standards module's more advanced features once they had become familiar with its general structure.

11.6.6 The Influence of Individual Characteristics

The study observed changes in the expectations and project contributions of the users and the developer linked to growing experience. The users' ability to communicate more complete concepts of products which might be developed grew with experience. At the same time, the developer's experience in both communicating with the users and applying the development tool aided the development process. These features highlight the changing environment over the course of a single development project. Future development projects involving a different mix of individuals and their working environment will change their relative starting points. A more computer literate group of users may hold more certain views concerning the form and function of the required product. A different developer may demonstrate greater or lesser skills in creating design solutions acceptable to the user group. The personalities within the group may influence the co-operation and communication amongst those involved. The organisation might place constraints on the project which either constrain the project or allow it to broaden further.

The DAVE project responded to an environment and a mix of individuals in which all of these factors had an influence on the outcome. The development philosophy of user centred design through prototype evolution provides a guide to accommodate these possible variations. User involvement within the design cycle establishes the stakeholders, allowing them to contribute at whatever level the individual chooses. A more experienced user may hold a more defined concept with which to influence the eventual design. Where concepts differ, the prototype provides a medium for communication to resolve differences. Whilst individual personalities could lead to non-participation or unconsidered rejection, the user focus of the design cycle provides a strategy for avoiding user alienation. The aptitude of the designer in applying the development tools and their creativity in design must influence the form of the product. However, the active involvement of the users and external experts allows the project to draw on the creative input of more than one mind. The DAVE project was able to manage the
constraints of the organisational environment in terms of the transfer of effort. Other potential constrains such as the freedom to select development tools and management approach to time attribution might have a greater impact on the implementation of this approach to development. However, small organisations are likely to have a greater potential for flexibility in addressing these issues.

11.7 Acceptance

![Diagram of Acceptance Phase]

Figure 11.6 The Acceptance Phase

Figure 11.6 presents the process model for the acceptance phase of the development cycle. This phase acted as a final quality control prior to delivery of the evolved prototype. The evolutionary approach to development changed the task from the traditional function of comparing the design against a detailed specification to confirming the acceptability of the product in terms of usability and functionality.
11.7.1 Module Evaluation

The 'modular system' lifecycle relied on design evolution through the prototype development cycle to advance from a minimal specification to a deliverable product. This approach shifted the project's emphasis from initial specification to multiple cycles of evaluation and iteration. The evolving prototype provided all concerned with an increasingly accurate representation of the final product. As a prototype developed, different issues were raised by its evaluation. The reported case studies demonstrated the transition of loose concepts towards tailored products. Large design changes were made to the initial non-functional prototypes. As the prototypes progressed, the changes related more to the form and function of the prototype's elements rather than with the application as a whole.

Figure 11.5 suggests the involvement of three groups in the module level evaluation of the evolving prototype, each providing a different perspective on the design. The users were best placed to assess the product in terms of its ease of use and applicability to their work programme. Their involvement was frequent due to the close contact of the environment but informal to reduce the project's burden on the regular work programme. The past development experience of expert users permitted assessment at a more technical level. Finally, the product's developer was responsible for balancing the views of these sources with the capabilities of the system and with individual design opinions ranging from module structure to the detailed form of its elements. The developer was well qualified to support this role, as a hybrid from both of these evaluation groups.

The risks of relying on voluntary user participation within the design cycle were considered to be the potential for disproportional influence and insufficient user feedback, (section 6.2). It was judged that these risks could not be removed without placing unsuitable demands on the user group. Instead, a formal acceptance phase was proposed to ensure that these factors did not lead to the delivery of an unacceptable product. The processes applied to evaluation of the final product over the course of the project included expert evaluation and formal user testing. The approaches applied are discussed below.
11.7.2 Expert Evaluation

Expert evaluation was applied to the lifecycle’s acceptance phase for the development of the first module. As the first deliverable product of the DAVE system, it was necessary to confirm that user centred evolution was able to achieve a product design acceptable to the group from the starting point of a minimal specification.

The expert’s development experience led to the assessment of the product in terms of accepted standards and guidelines for software design. The main advantages of expert evaluation at this point were the opportunity for gaining both an impartial opinion and the alternative perspective provided by the expert’s distance from the project. Whilst these factors helped to highlight a few design issues overlooked by the developer and the users, the recommendations focused on possible design enhancements as opposed to necessary changes, (section 6.7.3.1). Many of these enhancements were either considered unnecessary by the group or beyond the scope of the project. It was concluded that the relatively small contribution made by this process was a result of its position in the lifecycle. Expert evaluation was found to be more effective when applied as part of the development cycle, before the design was refined, (section 11.6.3).

11.7.3 Formal User Evaluation

Each DAVE module was developed in-house and was aimed specifically at the requirements of an accessible group of users. Final determination of the acceptability of the product depended on the group’s ability to apply the tool and their subjective responses to its use. A range of techniques were applied to determine these factors throughout the DAVE project. All of the formal user evaluations undertaken were based on user completion of a representative task using a fully functional prototype.

The prototype development cycle involved the users in the assessment of a semi-functional prototype throughout its development. The prototype’s transfer from the development phase marked the informal acceptance of each product by those involved in its development. In contrast, the acceptance phase sought to gather objective data concerning the suitability of the product for the task for which it was designed. In each case, this was supplemented by formally gathered subjective responses from the group.
As shown by figure 11.6, the acceptance phase was not intended to simply provide an official 'seal of approval', but instead to highlight possible design changes for implementation prior to delivery. This approach relied on the flexibility of the development tool to allow changes to the design at these late stages in the lifecycle. In each case the users were required to apply the prototype with minimal training. The intention was to reveal usability issues which training might hide. This "worst case" approach highlighted non-intuitive features such as the printing approach for the first module and searching problems encountered with the second module. Forcing errors at this stage allowed design problems to be addressed prior to delivery. This was considered to reduce the potential causes of user frustration with the software and by association, reduce the risk of rejection of the delivered product.

The existence of a functional prototype assisted the prototype development cycle, acting as a focus for discussion between the developer and the users during assessment of the design. The acceptance phase's evaluation processes for the final design continued to centre on the product. For each module's formal user test, the full user group completed a representative task using the functional prototype. By setting a standard task for all subjects it was possible to take measures that were comparable across the group. The measures taken for the various modules included completion times, accuracy, action sequences and subjective responses. The techniques used to gather these measures included background logging, direct observation, questionnaire completion and subject interview. The contribution of each of these measures to the product's evaluation depended on the characteristics of the product and the task it supported.

Background logging was included in all delivered in-house products. It provided the means of gathering usage records for each module after delivery. The use of background logging as the primary data gathering approach for formal user testing of the Anthropometry module was considered to be inappropriate for future user tests due to the level of effort required to analyse the results, (section 6.7.3.2). This was confirmed by the more effective alternative approach of direct observation used to record measures for the formal evaluation of the Standards module. Whilst background logging allowed user completion of the task in a truly representative environment, it was found that direct observation provided additional information concerning each user's approach to using the tool, (section 7.8.4). The observer was able to expand on
the issues raised by the session through subject interview immediately after completion
of the task. Direct observation was adopted as the preferred technique for formal user
testing. The advantages of direct observation and background logging were combined for
the evaluation of the Vision module to record the accuracy of data input, (section
8.10.6).

11.7.4 Usage Record

Background logging code was included as a standard for all delivered in-house products.
Its greatest contribution was as a method of recording performance measures for the
supplementary applications, (appendix H). In addition, the automatic logs provided by
each application after delivery provided an objective measure of their actual acceptance
by the user group, (section 10.2). The background logging code remained the same for
the modules throughout the DAVE development programme, ensuring that the results
were comparable for each module. This record revealed the trends in product use over
the study's time frame which allowed conclusions to be drawn relating to the relative
contribution of each product to the group's work programme.

The usage log was gathered and analysed towards the end of the DAVE project. Earlier use
of this information could have revealed declining trends in usage, indicating the need to
review and possibly upgrade a delivered product. This information would have provided
a useful additional information source for the task of system overview in the 'system
level' specification phase. This was not done in this project due to the effort associated
with the gathering and analysing the usage log files, as experienced in the acceptance
phase of the first module. However, its design could have been improved to automatically
compile usage reports to simplify the review process. Careful design of background
logging code, focusing on simple measures such as total module usage, could have
provided information to assist work programme selection.
11.8 Distribution and Implementation

![Diagram of the Delivery Phase]

Figure 11.7 The Delivery Phase

Figure 11.7 presents the process model for the delivery of the product at the end of each development or maintenance cycle. For the development cycle, delivery followed the formal evaluation within an acceptance phase. Delivery following the maintenance cycle relied on the results of past evaluation and the control of user input in the prototyping phase to ensure product acceptability.

11.8.1 Distribution for an Evolving Structure

The first module of the DAVE system followed a modification of the traditional lifecycle in which the end of prototype development was marked by acceptance and delivery of the final product, (figure 6.1). Because the system was to be presented on a standalone computer, the task of delivery was reduced to a single event at the end of the product's development cycle.
The growth in the group's computer resources responded to the computer industry's trend of increasing performance for decreasing cost. This corresponded with the organisation's trend towards increased computer utilisation. The increase in available equipment was taken by the DAVE project as an opportunity to boost module use by presenting its module on a series of desktop computers linked by an internal network. The increase in presentation platforms changed the development cycle's final task from delivery to distribution.

Whilst presenting the DAVE system on the desktop of each member of the user group increased the user convenience for accessing the modules, the levels of voluntary use did not show a sustained rise, (section 10.3.2). This was taken to indicate that the limiting factor for the use of the DAVE system modules was the actual work programme requirement for the information provided by each module as opposed to user reluctance to apply the module.

11.8.2 Post-Delivery Iteration

The lifecycle was amended following delivery of the DAVE system’s first module to allow the project’s limited resources to be returned to the development of the next module. The task of completing any minor corrections revealed by delivery of the product was allocated to an iterative cycle immediately following the product's delivery, (section 6.6). The larger task of design revision was not supported by the original sequential model. In anticipation of a demands for the upgrade of delivered products, a separate maintenance cycle was added to the ‘modular system’ lifecycle, (section 6.8).

Although the delivery phase's iterative process was not called on following the delivery of the first module, it was required for subsequent in-house products, (section 7.7.2). Design changes made after delivery needed to be provided across the user group. The relatively small distribution of the system allowed these changes to be implemented across the group by the developer whenever a necessary design alteration was raised. Systems with a wider distribution would require a formalised version control to ensure that design corrections were fully implemented across the user group.
11.8.3 Module Maintenance and Computer Support

The 'modular system' lifecycle’s alternative phases of development or maintenance enabled it to successfully manage the development and upgrade of its delivered modules despite its limited resources. Maintenance activities included the upgrade of the Anthropometry module and the spin-off development of variations on the Standards module. The demand for system upgrade was driven by a gradual shift in requirements, in line with rising standards in commercial software and hardware technology, (section 9.7.5).

The maintenance approach of the continued evolution of existing designs removed the need to design a prototype from first principles. This allowed the maintenance cycle to follow a simplified re-specification activity in place of the development cycle's more complete module level specification. Although starting from a more advanced prototype, the prototype development cycle continued to provide an appropriate structure for the product's iterative design, (section 9.7.2). The combination of user involvement and the template module's previous formal acceptance led to the rejection of formal evaluation within the maintenance cycle. The reduction of development effort, relative to the full development cycle, allowed incremental improvements to be made to the DAVE system at a level of effort in line with the product's added benefit, (section 10.6.2).

The task of computer support became an essential part of the developer's role with the expansion of the DAVE system and the group's increasing use of commercial software, (section 10.6.3). The project's response to this demand was to uncouple this activity from the lifecycle sequence which allowed an immediate response to requests for instruction and fault correction. Failure to support this activity would have risked user rejection of the in-house software products and compromised the efficiency of the group in the use of commercial software.
11.9 Conclusion

The development of the DAVE system provided the study with a series of case studies in which the issues surrounding the small scale in-house development were investigated and drawn together in a tailored lifecycle methodology. The project developed and maintained novel software applications which were used regularly by its target population in direct support for their specific work programme. In doing so, the study confirmed the feasibility of bespoke software development despite limited resources and user uncertainty. The study took an active research approach by guiding the conduct of the development project as well as evaluating its products. Over the course of the project a lifecycle methodology was designed and refined to meet the emerging characteristics of the project and its environment.

The study started from a point where little guidance was available relating directly to the characteristics of the project. It utilised the user group's accessibility to counteract their uncertainty through a user centred evolutionary approach to development. New products started from a minimal specification which reflected the user's level of certainty concerning their requirements and the form of a solution. Prototype development provided an accessible medium for the expression of designs and establishment of a common concept for all concerned in its design and evaluation. The study's prototype development cycle was used successfully to advance and refine the design of a series of modules. Prototype evolution provided the means of bringing the prototype and the user group's concept progressively closer, leading to products which were both usable and used. Formal evaluation of the fully functional prototypes ensured acceptability prior to their delivery as a system module. This evolutionary approach to development was used to maintain and expand the DAVE system to meet the changing requirements of the users and their organisation.

This evolutionary approach to development was set out in the study's 'modular system' lifecycle methodology, (figure 11.8). This model draws together the cumulative findings of the DAVE project case studies to provide a comprehensive methodology equipped to manage all aspects of a small scale in-house development project. The lifecycle proposes successive development of system modules to allow an individual developer to build and maintain an expanding software tool. The lifecycle is separated
into a series of phases for which process models are set out. User centred review of the
project directs the project's continuing work programme without finely defining its
concept product. The evolution of the concept's design leading to a deliverable module is
conducted within a 'module level' development cycle. An alternative 'module level'
maintenance cycle offers a reduced procedure for the modification of existing modules.
The development or maintenance cycles return to a system level review of the project,
anticipating a change in requirements over time. Finally, the lifecycle includes a
separate computer support activity to meet user demands for instruction and fault
correction.

The 'modular system' lifecycle draws together the lessons of a longitudinal study of a
parallel development project. The resulting methodology provides a development
strategy which proficiently applies the advantages of the in-house environment to
overcome its constraints.
Work Programme Specification

Background Analysis
- Analysis of Requirements
- Platform & Tool Selection

Overview
- Developer Review
- User Interview

Specification
- Formal Work Programme Selection

DEVELOP

SYSTEM

M O D U L E

Design Specification
- Data Set Selection
- Requirements Definition
- Functional List Construction

Prototyping
- Specify
- Design
- Evaluate

Acceptance
- Formal User Testing
- Alteration

Delivery
- Distribution/Implementation
- Alteration

ELEMENT

A

B

C

Evaluate

1. Developer
2. User Interview
3. Expert

Figure 11.8 The 'Modular System' Lifecycle
11.10 The Role of the 'Modular System' Lifecycle

In chapter 3, a review of the literature relating to small scale in-house development was presented. It reported the promised scope of new development tools but was unable to establish adequate guidance for their implementation for this important category of software development. The guidance available included detailed methodologies and the less prescriptive approach of lifecycle models.

Detailed formal methodologies such as SSADM were rejected at an early stage as they were considered to demand resources beyond the scope of the small organisation. Their focus was the development of large complex systems for which they provided an established method of managing product development. The limited spread of these methods within their intended class of development, (Raghavan and Chand 1989), provided an initial warning against their application to small scale development. Beyond the fundamental limitation of the available resources, they did not match the study scenario’s characteristics of an uncertain product concept and access to a user population of a manageable size. In contrast the proposed 'modular system' lifecycle provides a framework that offers appropriate management of the limited resource of the individual developer and an evolutionary strategy to build user understanding along side the evolving prototype.

The waterfall model, (Royce 1987), provided a traditional view on development. Its linear approach was considered to match development projects with concise definitions of the required product, (James 1991, Harker 1992). The model’s clear structure offered a starting point from which the 'modular system' lifecycle emerged through its modification to meet the determined features of the study's scenario as they became apparent. The central issues encountered were the absence of user certainty in product definition and the instability of specifications. Changes in a user group’s work programme, working practices and technological options demand flexibility from a methodology to avoid inappropriate product delivery and to ensure continued support through adequate maintenance. By moving from this traditional linear sequence to the twin cycles of development and maintenance, the 'modular system' lifecycle provides both the opportunity for revisiting the group’s requirements and a model for incorporating these changes within an emerging modular product.
The spiral model, (Boehm 1988), offered a strategy for addressing the inherent uncertainty in software specification and design through a spiralling set of activities. The model was characterised by the iteration of a sequence of prototypes as a precursor to detailed design, coding and eventual delivery. In doing so, it provides a series of milestones ideally suited to formal team meetings and occasional user contact, consistent with the lifecycle's focus on large scale development. In contrast the 'modular system' lifecycle builds on the study scenario's key feature of continual close contact between the developer and the users by increasing the points of user involvement to the design of a gradually evolving product.

The rapid prototyping model suggested by Howell (1992) demonstrated the benefits of modifying a lifecycle's structure to meet a particular scenario. By integrating rapid prototyping within a linear process, it offered a development approach which was tailored to a particular scenario. Design of the 'modular system' lifecycle followed this example by modifying the waterfall model to support the characteristics of small scale in-house development. The resulting methodology provided a framework capable of accommodating the resource limitation of an individual developer through its twin cycles of development and maintenance. The principle of evolutionary development from a minimal specification is set out within a defined prototype development cycle, at the core of the lifecycle. The result is a lifecycle methodology that builds on the positive characteristics of the scenario to overcome its limitations.

Whilst the 'modular system' lifecycle was designed and evaluated within of a series of case studies, its implications for future development spread beyond the particular circumstances of the study's scenario. In exploring the lifecycle's design, broad characteristics such as user group size, experience, mutual accessibility and confidence in specification were raised as features for which it must account. Limiting features such as available resources and user uncertainty were countered by the methodology's formalised cycles of development and maintenance based on the strategy of evolutionary development from a minimal specification. By demonstrating the success of this approach through product development and implementation, the study provides a formal strategy for other development projects faced with similar characteristics.
11.11 Support for the Study's Aims

The study adopted an action research approach through which a guide to software development was iteratively tailored to the characteristics of its scenario. This approach provided the means of addressing a range of issues, which were grounded in the practicalities of small scale in-house software development, drawing them together in a formal lifecycle methodology. The study must be judged finally by its contribution to the central arguments of the thesis, (section 1.3). Conclusions drawn and supporting evidence provided for these arguments are reviewed below.

1. Evolutionary prototyping is a feasible approach for small scale in-house software development.

The approach of evolutionary prototyping for small scale in-house development was investigated within a series of case studies. Each reported the successful evolution of a design concept from a non-functional prototype into a deliverable application, bypassing the traditional coding phase of software development. The study combined the flexibility of high level development with the in-house environment's scope for direct interaction between the developer and the user to overcome the limitations of the study's scenario. It demonstrated the ability of user centred exploration and refinement of a common concept through prototype iteration to effectively counter stakeholder inexperience, uncertainty in specification and changing requirements over time. In doing so, it confirmed the feasibility of evolutionary prototyping for small scale in-house software development.

2. Design evolution from a minimal specification based on direct user involvement is an appropriate approach for the development and delivery of an acceptable product.

The study proposed a strategy of user centred design evolution based on a minimal specification as an alternative to detailed specification prior to design and coding. This approach relied on close user involvement to guide prototype evolution from a minimal concept to an implemented application. It was successful in overcoming
user uncertainty by the progressive development of the concept as a model upon which the users could consider issues of design and content. Prototype evolution based on high level development allowed a balance to be met between creative design and user control to achieve novel applications that were shown to be usable by formal evaluation and shown to be acceptable to its target population by achieving consistent levels of usage.

3. A new lifecycle methodology can be tailored to the characteristics of small scale in-house software development as a guide to subsequent projects.

The prescriptive approach of formal methods was rejected in favour of tailoring a lifecycle methodology to the characteristics of the study's scenario. The features of a small accessible user group, development by an individual and user uncertainty were not adequately addressed by those lifecycles available at the start of the study. The study's action research approach planned, implemented and evaluated the 'modular system' lifecycle during its evolution and validated it as a guiding methodology for the conduct of small scale in-house software development.

11.12 Limitations of the Study

The study was carried out as a parallel activity to the software development project. It adapted to a naturalistic approach which ensured the consideration of the system within the context of its operational environment. Whilst offering the most appropriate approach to investigation, it brings with it limitations. These are considered to be as follows:

- The size of the user group.

Access to the products of the DAVE development project was restricted to the specified user group. This small number of users places a limit on the generalisation of conclusions drawn concerning the suitability of the products for a wider user population. However, the lifecycle and its constituent processes offers guidance to future development projects able to relate their scenario to the reported characteristics of the DAVE project.
• Lack of control within the organisational environment.

Whilst the development programme was guided by the processes set out by the parallel study, overall control remained with the customer group. This led to deviations from the ordered lifecycle sequence. Examples include the overlap of two projects and the demand for a shift in developer effort to the development of supplementary applications in support of specific work programme tasks. Although this reduced the study's ability to provide closely comparable development projects these demands were a consistent feature of the environment. Consequently, the strategies adopted to meet these demands provide guidance for similar projects.

• Changes in individual experience over the project time-frame.

Although the observed rise in the experience of the user group and the developer over time could be expected to influence similar development projects, this change in the character of information feedback to the cycle must be taken into account when comparing the relative success of the sequential development and maintenance projects.

• Limited comparison against other methods.

The study was required to provide the most appropriate approach for each project. It started by extracting the principles considered to best support the project. This led to iterative tailoring of a lifecycle to match the emerging features of the project. With the exception of the comparative implementation of the HUFIT toolset for specification, the formal methods available were considered unsuited to the project's characteristics.
11.13 Recommendations for Further Research

The 'modular system' lifecycle provides a methodology to manage the full range of demands placed on the development project. The approach of evolutionary design based on minimal specification closely matched the key features of the project. These features included accessibility of the full user group, absence of a firm product concept and the limited resources of an individual developer. Further research is required to support and expand the findings of this study. Suitable areas for future work are as follows:

- Further comparison of the lifecycle's component processes.

The selection of the processes for each lifecycle phase was based on the assessed objectives of that point in the cycle. The techniques and tools applied to meet these goals were contrasted to provide recommended process models. However, the study was unable to exhaustively compare available techniques for each phase. Further comparison of techniques to meet the characteristics of the development environment could improve the efficiency of the lifecycle for in-house development.

- The detailed measurement of the effort required for evolutionary development.

The project constraints prevented adequate evaluation of the relative efficiency of design exploration as opposed to detailed specification as a development strategy. The success of the DAVE project's in-house products confirmed the ability of design evolution to extract and meet user requirements. Although developer time allocation records provided a measure of associated cost, a more detailed account of effort for all involved is required. Such a measure could be extended as the basis of a metric for predicting project cost within the lifecycle's 'system level' specification phase.

- A survey of software development for small user groups.

At the start of this study, the development of small scale software was largely unreported. This project demonstrated the feasibility of in-house software development, tailored to the requirements of a small user group. Over the study's time-frame, technology has improved to simplify the task of software development further.
Given these advances, a survey of small scale software development could provide an indication of the prevalence of this technique and reveal the methods adopted currently by this population of developers.

- The determination of a metric to mark the point of completion of the prototype development cycle.

Whilst the risk of prototype over development was recognised and tackled within this project, the problem was not finally resolved. Determining the appropriate point of transfer between evolutionary development and the lifecycle’s formal acceptance phase may benefit from the determination of a metric to formally mark prototype completion. An objective calculation balancing the cost of continuing development effort with the anticipated benefit might be able to improve the effective allocation of effort within the ‘modular system’ lifecycle.

- A review of multi-platform solutions to information retrieval and presentation.

The project was required to choose a dedicated platform for the development and presentation of the product. This prevented the delivery of the DAVE system to the organisation’s wider population following the transition to a larger centralised human sciences research centre. Whilst the apparent convergence in the computer industry may eventually remove this distinction, investigation of cross platform development tools for the purpose of evolutionary prototyping could lead to the wider application of the products developed following the recommended lifecycle.

- Application of the lifecycle to larger development projects.

The ‘modular system’ lifecycle was tailored to the characteristics of the small scale in-house development project. The lifecycle may be applicable to the development of modular systems on a larger scale. The lifecycle structure provides a project management approach which manages all stages of a modular system’s life. Expanding the size of the development team or the number of users would require a re-evaluation of the structure and the re-selection of its component processes. In particular this expansion would need to address the role of the users or their representatives in the evolutionary design of a software product intended for wider distribution.
• A review of advances in software development tools.

The five year period preceding the DAVE project witnessed the emergence of the desktop computer. Within the study's time-frame the desktop computer spread to dominate organisational practice and reach into the home. This has been linked to a shift to the graphical interface as the accepted standard for presentation and interaction. The next five years and beyond promise further advances in the areas of processing efficiency and cost, among others. The implications of these changes on the particular niche of the small user group require further consideration. As the volume of multi-modal information available grows, it is important to consider how this data is to be managed and the role of the individual or the small scale developer in tackling this task.
References
ADOBE SYSTEMS, INC. (1989) Photoshop™, Version 2.0

APPLE COMPUTER, INC. (1989) HyperCard®, Version 1.2.5


BEAGLEY N.I., (1996a) Field Based Prototyping, Usability Evaluation in Industry, Jordan, Thomas, Weedmeester and McClelland (eds) Ch. 11, pp. 95-104


323


BLYTHE SOFTWARE Ltd. (1989) Omnis 5, Version 1.2


327


HMSO (1993) Software Lifecycle Support, CTTA, *IT Infrastructure Library*


JENKINS, J (1993) File on Four Transcript 19/10/93 (19:20) Programme No. 93VY3042LHO


329


330


MINISTRY OF DEFENCE, 1983-1996, Defence Standard 00-25, Issue 1, Parts 1-13


NEOW Ltd. (1990) Toolbook, Version 1.5


SILICON BEACH SOFTWARE, INC. (1989) SuperCard™, Version 1.6,


334
SYMANTEC CORPORATION (1989) Think C™, Version 4.0


Considered as a Craft, Taking Software Design Seriously, Practical Techniques for
Human-Computer Interaction Design, Karat (ed), pp. 1-19

YOUNG, D., LANSDALE, M. W. and BASS, C. A. (1990) Using HyperTalk as a
Specification Tool and a Simulation Vehicle in the Development of a Personal Data
Base System, Simulation and the User Interface, Hamilton (eds), (Taylor and
Francis), pp. 169-180
Appendices
Appendix A

Initial User Interview

Interview Sheets

Date -

Subject -

1 - What would you consider to be your main areas of work?

   Assessment - What of?
   Report writing - What subject?
   Form filling - Which forms?
   Information search - what Info.?
   where from?

   Other - Please specify.

2 - Have you used a computer database in the past?

   Yes - Which system? What were your impressions?

   No - Why?
3 - What do you understand by “An OPC Knowledge Base System”?

What should it include for The SECTION?

What should it include for YOU?

4 - What would make you want to use it?

5 - What would stop you from using it?

6 - In what form do you prefer information to be presented?
   - Text / Pictures / Combination
   - Other (e.g. Speech)

7 - Do you feel comfortable using a Keyboard
   - Mouse
   - Touchscreen

8 - Which do you prefer to read from a Computer Screen / Paper?
   - Why?

9 - Any other points?
Appendix B

**Anthropometry Module Evaluation Task**

In order to evaluate the Anthropometric database I need to enlist your help. I have prepared a set of questions which I would like you to please work through by using the anthropometry database. Please put the numerical answers in the brackets at the end of each question. A space has been left after each question for any comments you might have. If you are unable to complete any of the questions please describe why. The aim of this sheet is to provide you with experience in using the system and to provide me with the feedback necessary to improve the system.

Name .................................................. 
Date .................................................. 
Time .................................................. 

1) Please open the anthropometry screen

   *Comments (if any)*

2) What is the Stature of a 95th Percentile Guardsman - [ .................]

   Please tick if you
   a/ Visually searched the lists for the measure [ ]
   b/ Placed the cursor over the photograph hot-spot to highlight the list measure [ ]

   *Comments (if any)*

3) What is the Functional Reach of the 5th %ile Guardsman - [ .................]

   Please tick if you
   a/ Searched the lists [ ]
   b/ Used the photograph hot-spot [ ]

   *Comments (if any)***
4) What is the Waist Circumference of the 50th %ile Guardsman
   
a/ in Cm - [....................]
b/ in Inches - [....................]

Comments (if any)

5) What is the Weight in Kg of a 79th %ile Guardsman - [....................]

Comments (if any)

6) What proportion of Guardsmen have a bideltoid breadth of less than 48 cm - [....................]
   - How did you find this value

Comments (if any)

7) Printout the 5th & 95th %ile Guardsman values relating to hand and foot dimensions
   Please tick if you
   a/ Searched the lists
   b/ Used the photograph hot-spots [ ]

Comments (if any)

8) What is the 99th %ile Shoulder Height of a member of the R.A.C. - [....................]

Comments (if any)
9) What proportion of subjects in a combined survey of Guards, R.A.C. & Royal Artillery can squeeze into a crew-station if the seat squab to hatch distance is 93.5 cm?

Comments (if any)

To what accuracy has the weight measures been taken

Comments (if any)

Please fill in below any other comments you have concerning the interface, functionality, etc. of the system.
Appendix C

Electronic File Module Evaluation Task

The Electronic File module of the DAVE system provides access to selected ergonomic reference data through indexed text and photographic representations of each of the document's pages. The following task compares the ability of the computer system to answer ergonomic questions with the current paper based method. The following task involves answering six questions using Def. Stan. 0025 as reference document. The experimenter will demonstrate two examples of finding references using either the Electronic File or the Paper Document. Following the demonstration please answer the first three questions on the sheet using the Electronic File module. Use the Paper document to answer the remaining three questions. After each answer please indicate the section and page number of the reference. As you carry out the task try to vocalise your actions explaining, where possible, the strategy you are adopting. If you encounter any problems when using the computer system, try to answer these using the built in "help" system. The "help" system is activated by pressing the button in the top right hand corner of the screen. Following completion of the task there will be a short interview concerning the use of the system and the task of information retrieval.

Demonstration Questions

i  What size computer keys are needed for a QWERTY keyboard for use in a low level vibrating environment?
   A ................................................................................................ ( / )

ii At what temperature does human tissue burn?
   A ................................................................................................ ( / )

Test Questions

Computer

1 What level of illuminance is required for the interior of a command vehicle used for map reading at night?
   A ................................................................................................ ( / )

2 What height adjustment should be provided for a chair?
   A ................................................................................................ ( / )

3 At what knee angle is maximal force applied by the leg?
   A ................................................................................................ ( / )

Paper

4 Is there a standard that addresses hand vibration?
   A ................................................................................................ ( / )

5 What is the minimum size for an escape hatch?
   A ................................................................................................ ( / )

6 How much space do you need to leave between two toggle switches allowing for NBC IPE?
   A ................................................................................................ ( / )

C1
The following tables were completed following the structure laid out by the PAS tools of the HUFIT toolset. They were completed through user interview as the means for specifying the Vision module leading to design and construction of prototype B.

<table>
<thead>
<tr>
<th>User Groups</th>
<th>User Goals</th>
<th>Benefits</th>
<th>Costs</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPC</td>
<td>To input necessary information</td>
<td>Information handling capabilities of IT are improvement over existing manual editing and storing using paper.</td>
<td>Require portable IT product with power source.</td>
</tr>
<tr>
<td></td>
<td>To view plots</td>
<td>Instant construction of vision plots; much quicker compared to existing pen/paper method. Allows improved OPC contribution in the post assessment &quot;wash-up&quot; meeting.</td>
<td>Require portable printer to obtain hard copy of plots when in the field, or access to a compatible printer.</td>
</tr>
</tbody>
</table>

Table D.1 Background details

<table>
<thead>
<tr>
<th>User Groups</th>
<th>Description</th>
<th>Product Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPC</td>
<td>Age/Gender</td>
<td>Good size typeface</td>
</tr>
<tr>
<td></td>
<td>20-45 Both genders</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Education</td>
<td>Knowledge of scientific and task terms assumed, within reason. Assume knowledge of relevant Army systems</td>
</tr>
<tr>
<td></td>
<td>Degree/M.Sc. level</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Minimum 3 years experience of Army vehicles and systems.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Other</td>
<td>Computer Literate</td>
</tr>
</tbody>
</table>

Table D.2 User Group Characteristics
Table D.3 Use Characteristics

<table>
<thead>
<tr>
<th>User Groups</th>
<th>User Characteristics</th>
<th>Product Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPC</td>
<td>Frequency of Use</td>
<td>Easy to recall procedures</td>
</tr>
<tr>
<td></td>
<td>Low frequency(&quot;3 per annum)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Discretion to use</td>
<td>Copy existing task elements into interface procedures. Stress usability of HCI</td>
</tr>
<tr>
<td></td>
<td>Low - Assume all future vision assessments to use application tool</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Task Knowledge</td>
<td>Provide flexible task completion format and &quot;on-line&quot; help facility.</td>
</tr>
<tr>
<td></td>
<td>Existing users = High. Assume new users have task knowledge</td>
<td></td>
</tr>
<tr>
<td></td>
<td>IT Knowledge</td>
<td>Match stereotype HCI conventions found in existing software products used by OPC</td>
</tr>
<tr>
<td></td>
<td>Moderate - High</td>
<td></td>
</tr>
</tbody>
</table>

Table D.4 Task Environment

<table>
<thead>
<tr>
<th>Task</th>
<th>Product Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location: Inside a military vehicle</td>
<td>Portable computer and power source.</td>
</tr>
<tr>
<td>Workspace: Crew workstation</td>
<td>Self-illuminating screen with simple input operations.</td>
</tr>
<tr>
<td>Lighting: Daylight/small directional spotlight</td>
<td>Can use auditory cues/error message indicators.</td>
</tr>
<tr>
<td>Noise: Negligible</td>
<td>Input procedure to allow user to wear gloves</td>
</tr>
<tr>
<td>Other: Potentially cold environment as field based and static.</td>
<td></td>
</tr>
</tbody>
</table>

Working Conditions

Work with two other assessors; directly communicate with one using hand-held radio.

Interruptions: On-going as plot positions estimated. Also, to change power source.

Hold in position with one hand.

Make clear user position in procedure and that inputs regularly saved.
<table>
<thead>
<tr>
<th>User group</th>
<th>Task Characteristics</th>
<th>Product Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPC</td>
<td><strong>Variability of Input/Output</strong>&lt;br&gt;May have to avoid 1+ input variables.&lt;br&gt;Need to alter input data.&lt;br&gt;Variable screen outputs</td>
<td>Facility for avoiding input variables, therefore altering data easily. Graphic &amp; tabular output</td>
</tr>
<tr>
<td></td>
<td><strong>Multi-Task Demands</strong>&lt;br&gt;H.C.I adjacent to positioning plot marker using hand-held radio.</td>
<td>Must make explicit user whereabouts in the system</td>
</tr>
<tr>
<td></td>
<td><strong>External Pacing</strong>&lt;br&gt;None</td>
<td>Permit freedom to move between task elements.</td>
</tr>
<tr>
<td></td>
<td><strong>Autonomy</strong>&lt;br&gt;Moderate, once vehicle information inputted i.e. freedom to choose which vision devices/workstations to assess.</td>
<td>As above.</td>
</tr>
<tr>
<td></td>
<td><strong>Formal Structure</strong>&lt;br&gt;No set end/start to task of plot data input.</td>
<td>As above.</td>
</tr>
<tr>
<td></td>
<td><strong>Error Tolerance</strong>&lt;br&gt;No tolerance; incorrect inputs could have severe consequences-commercial, strategic and political.</td>
<td>Ensure accuracy of input; easy updating facilities; good error handling and correction.</td>
</tr>
<tr>
<td></td>
<td><strong>Confidentiality</strong>&lt;br&gt;Classified information</td>
<td>Use password to gain access to data. Store all data files on floppy disk-separate to program. Allows data and computer to carried separately to/from assessment sites.</td>
</tr>
</tbody>
</table>

Table D.5 Task Characteristics
<table>
<thead>
<tr>
<th>Design Area</th>
<th>What is required</th>
<th>How it is required</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input devices</td>
<td>Compact/Portable/Powerful/Robust. Flexibility</td>
<td>Use Macintosh PowerBook computer which has trackball as an alternative input device.</td>
</tr>
<tr>
<td>Displays</td>
<td>Clarity between screen segments; self illuminating.</td>
<td>Available PowerBook has black &amp; white screen, therefore choose contrast shades carefully.</td>
</tr>
<tr>
<td>Output Devices</td>
<td>Graphical plots and tabular summary information. All associated data inputs to be stored in a file.</td>
<td>Output screen split into graphic and table areas</td>
</tr>
<tr>
<td></td>
<td>Specify plot criteria. Layout of output screen to suit A4 portrait paper orientation(for report writing)</td>
<td>File management system</td>
</tr>
<tr>
<td>Printers</td>
<td>Facility to produce “hard copies” of plots.</td>
<td>Access to compatible(on-site) printer, or provide portable self-powered type.</td>
</tr>
<tr>
<td>Connections</td>
<td>Transfer plot outputs to other software packages</td>
<td>Ensure plots can be saved in standard format and/or exported as graphics.</td>
</tr>
<tr>
<td>Storage</td>
<td>Data separated from computer</td>
<td>Floppy diskettes.</td>
</tr>
<tr>
<td>Software</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Styles and features</td>
<td>User interface that is consistent, flexible, navigable, has good layout, simple to use, tolerant, efficient, helpful and natural.</td>
<td>Graphical user interface Variety of input methods Broad and shallow structure Software features that are consistent with existing stereotypes used in current OPC software applications. Help system and error handling capability(with respect to accuracy of data inputs/outputs)</td>
</tr>
<tr>
<td>Training and</td>
<td>Minimal</td>
<td>One training session. Use on-line help.</td>
</tr>
<tr>
<td>Documentation</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table D.6 User Requirements Summary
## Appendix E

### HUFIT Functionality Matrix (Prototype B)

<table>
<thead>
<tr>
<th>Functionality Specification - Prototype B</th>
<th>Hardware</th>
<th>Basic Functions</th>
<th>Software Interface</th>
<th>Add. Func.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**EST. DEVELOPMENT TIME**

| Task Characteristics | Use in a task environment | Use with communication device | Use with a display | Continuous display | Delayed display | Delayed data input and output | Frequent changing data entry | Frequent changing data category | Fast paced | Different levels of info. req. | Low variation in task | Basic sequence of operations | Quick access to other displays | Frequent user entry errors likely | Frequent user int. actions | Few new actions needed | High anxiety | White audience to view output | Under-other task affiliation | User Characteristics | Compatitive task - high semantic - low syntactic | Low frequency of use | No data entered | Application must be used in future | Experts and last access in data | High intelligence | Experience of Supervised applications | Educated variable | Low variation in task | Sub-tot - User / Task / Functionality | Final Assessment - Functionality |
|----------------------|--------------------------|----------------------------|--------------------|-------------------|-----------------|-----------------------------|-----------------------------|----------------------------|----------------|---------------------------|-----------------|-----------------------------|-----------------------------|---------------------------|-------------------|----------------|-----------------------------|-----------------------------|----------------|-----------------------------|-----------------|----------------|-----------------------------|-----------------------------|----------------|-----------------------------|-----------------|----------------|-----------------------------|-----------------------------|----------------|-----------------------------|
Appendix F

Vision Module Evaluation (Prototypes A & B)

Usability evaluation

Subject No:

The following questions will ask you to compare Prototype A to Prototype B. The purpose of the questionnaire is to find out which application you prefer to use, (Prototype A or Prototype B) however this does not prevent you from suggesting features of both that you would like to see incorporated into a future version. Please provide a short written answer to the following questions, unless stated.

Instructions for Qu.1: Can you compare vehicle information and plot measurement screens between applications and circle which was better on the criteria being examined. In addition, can you briefly specify what it was that was better about the preferred prototype, and worse about the other prototype that lead you to your conclusions

1. Was it simple to read the information presented on each screen? (Information should be clear, well organised and easy to read)

Prototype A:
Vehicle information
Plot Measurement

Prototype B:
Vehicle information
Plot Measurement

Prototype A and Prototype B:
View Plots
Qu. 2 -10: Can you circle which was the better prototype on the criterion being examined, and briefly explain why it was better and why the second prototype was worse.

2. Were the features of each application consistent between screens e.g. abbreviations, pictorial information, selecting options, data entry. (The way each application works should be consistent at all times)

Prototype A

Prototype B

3. Were the features of each application compatible with the task and user conventions of other software packages that you use?

Prototype A

Prototype B

4. Was the structure of each application obvious to you? (The way each works and is structured should be clear)

Prototype A

Prototype B

5. How good was each application at providing informative feedback? (Each application should give clear feedback on what actions you have taken, whether they were successful and what actions you should take next)

Prototype A

Prototype B
6. Did each application possess adequate functionality (including Help facilities) to meet the needs of the task - collecting vehicle information, inputting plot data and constructing vision plots? (The needs and requirements of users when carrying out the tasks should be met)

Prototype A

Prototype B

7. Were the interfaces of each application flexible? (They should be flexible in the way they present information and how they allow you to complete the task, so that you feel in control)

Prototype A

Prototype B

8. Which application had the best aesthetic design overall?

Prototype A

Prototype B

9. Was the Powerbook easy enough to use on your lap? (take into account the different data input methods)

Prototype A

Prototype B

10. Did you experience any other usability problems with either application?

Prototype A

Prototype B
11. What were the best features of Prototype A?

12. What were the best features of Prototype B?

13. What were the worst features of Prototype A?

14. What were the worst features of Prototype B?

15. Which application would you most like to use in future given that any shortcomings you found could be improved to meet your requirements?

   Prototype A   Prototype B

16. Were there any features of the non-preferred application that you would like to see incorporated into an improved system?

   Thank you for completing this questionnaire.
Appendix G

Vision Module Evaluation (Prototype C)

Usability evaluation

Please circle the numbers which most appropriately reflect your impressions about using this application. There is room on the last page for your written comment.

PART 1: Overall user reaction

1.1 Overall reactions to the system: terrible wonderful
   1 2 3 4 5 6 7 8 9

1.2 frustrating satisfying
   1 2 3 4 5 6 7 8 9

1.3 dull stimulating
   1 2 3 4 5 6 7 8 9

1.4 difficult easy
   1 2 3 4 5 6 7 8 9

1.5 inadequate power adequate
   1 2 3 4 5 6 7 8 9

1.6 rigid flexible
   1 2 3 4 5 6 7 8 9

PART 2. Screen

2.1 Were the screen layouts helpful? never always
   1 2 3 4 5 6 7 8 9

2.2 Was the highlighting on the screen helpful? not at all very much
   1 2 3 4 5 6 7 8 9

2.3 Beginning, middle and end of tasks confusing clearly marked
   1 2 3 4 5 6 7 8 9
PART 3: Terminology and System Information

3.0 Messages which appear on the screen

<table>
<thead>
<tr>
<th>confusing</th>
<th>clear</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 3 4 5 6 7 8 9</td>
<td></td>
</tr>
</tbody>
</table>

3.1 Does the computer keep you informed about what it is doing?

<table>
<thead>
<tr>
<th>never</th>
<th>always</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 3 4 5 6 7 8 9</td>
<td></td>
</tr>
</tbody>
</table>

3.2 Does the terminology relate well to the work you are doing?

<table>
<thead>
<tr>
<th>unrelated</th>
<th>well related</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 3 4 5 6 7 8 9</td>
<td></td>
</tr>
</tbody>
</table>

3.2 Error messages

<table>
<thead>
<tr>
<th>unhelpful</th>
<th>helpful</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 3 4 5 6 7 8 9</td>
<td></td>
</tr>
</tbody>
</table>

PART 4: Learning

4.1 Learning to operate the system

<table>
<thead>
<tr>
<th>unhelpful</th>
<th>helpful</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 3 4 5 6 7 8 9</td>
<td></td>
</tr>
</tbody>
</table>

4.2 Exploration of features by trial error

<table>
<thead>
<tr>
<th>discouraging</th>
<th>encouraging</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 3 4 5 6 7 8 9</td>
<td></td>
</tr>
</tbody>
</table>

4.3 Remembering names and use of commands

<table>
<thead>
<tr>
<th>difficult</th>
<th>easy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 3 4 5 6 7 8 9</td>
<td></td>
</tr>
</tbody>
</table>

4.4 Can tasks be performed in a straight-forward manner?

<table>
<thead>
<tr>
<th>never</th>
<th>always</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 3 4 5 6 7 8 9</td>
<td></td>
</tr>
</tbody>
</table>

PART 5: System Capabilities

5.1 System speed

<table>
<thead>
<tr>
<th>too slow</th>
<th>fast enough</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 3 4 5 6 7 8 9</td>
<td></td>
</tr>
</tbody>
</table>

5.2 How reliable is the system?

<table>
<thead>
<tr>
<th>very unreliable</th>
<th>very reliable</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 3 4 5 6 7 8 9</td>
<td></td>
</tr>
</tbody>
</table>

5.3 Correcting your mistakes

<table>
<thead>
<tr>
<th>difficult</th>
<th>easy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 3 4 5 6 7 8 9</td>
<td></td>
</tr>
</tbody>
</table>

G2
Part 6: User clarification.

6.1 If you circled a five, or less in response to any of the questions could you please briefly state what your reasons were for doing so.

Part 7: Further Development

7.1 If the application was delivered in its present form, is there any aspect of its use that you would rate as unacceptable?

7.2 Before the application is delivered would you like to see any improvements made to it? If you would, can you state whether these improvements are 'essential', or 'desired'.

PART 8: User's comments

8.1 Please write any other comments you may have regarding the use of this application.
Appendix H

Supplementary Software

The three software applications described in the following sections were produced as a supplement to the DAVE system development project. The emergence of a software development capability within the group and the management autonomy provided by the organisation's structure allowed development effort to be occasionally diverted from the main development programme. This led to the development of several software applications and utilities designed to meet immediate work programme demands. In each case presented, the development project was constrained by a tight development time-scale and an inaccessible user population. Whilst following an evolutionary prototyping approach centring on the prototype development cycle, (figure 5.6), they did not follow the procedural restrictions imposed on the DAVE system's development projects.

A. The Panel Prototyper

![Figure H.1 The Panel Prototyper](image)

The concept of a reconfigurable hardware panel prototyper arose from the Interface test application produced at the start of the project. Further development of the concept was not undertaken within the main project as it did not fit within the group's concept of the DAVE system as an information retrieval tool. However, the concept was resurrected as a teaching tool in preparation for a student visit hosted by the group.
The form of the Panel Prototyper followed a smooth evolution from the design of the Interface application, (figure 5.5). The developer was given full control over design decisions to allow delivery of the tool within the tight time-scale. Design additions included the provision of a component palette, free positioning of components and linking e.g. a switch turning a panel light on or off. These were added to the application and designed following the main project's prototype development cycle with the absence of user input.

A tutorial was devised requiring the visiting student group to design and evaluate a control panel for a military vehicle. In a meeting at the end of the week, the students were required to report on their various tutorial sessions. The varied panel designs produced by the separate tutorial groups illustrated some of the alternatives to a single design problem and demonstrated the value of prototype for exploring design solutions.

Further feedback from the tutorial groups raised usability problems remaining in the Panel Prototyper. Further problems were revealed by observation of the groups completing the tutorial tasks. Problems encountered included component linking and a confusion between interaction modes. As a consequence of these problems it was necessary for the developer to support each tutorial group for the process of building their panel design.

These usability issues were addressed in a redesign of the Panel Prototyper for a return visit of the equivalent student group two years later. The revised design, (figure H.1) allowed the tutorial groups to complete the same exercise without developer support. This illustrates the important contribution played by user feedback to the design of a usable product. Whilst the first version of the Panel Prototyper met its functional aims the iterated design allowed the tutorial groups to concentrate on the task of panel design rather than operation of the prototyping tool.
B. The Map Simulation

Figure H.2 The Map Simulation

As noted for the Panel Prototyper the group concept of the DAVE system as an information support tool excluded some product concepts from recognition under the main stream project. One of the most important of these concepts in terms of contribution to the group's main work programme was the Map Simulation. The group's research work was focused on assessing a future vehicle concept through simulation in the group's mobile reconfigurable crewstation (Streets, et. al. 1993). An approximation of the concept vehicle's crewstation was constructed in a Fiberglas hull and placed on the back of a truck. The research programme aimed to highlight human factors issues relating to vehicle operation by testing the crewstation using representative crew within a field environment.

Meeting the crewstation concept required the inclusion of new tools anticipated for inclusion in future vehicle designs such as a mast mounted video surveillance system and an advance navigation system. Whilst it was possible to acquire most of the required systems from commercial suppliers, available computer based map systems were not sufficiently advanced for the purposes of the vehicle simulation. The solution was to develop a simulation of a future map through the groups new software development capability.
Development of the DAVE system software was sidelined in favour of development of a computer based map simulation to meet an impending field trial date. Development followed a more formal approach than the Panel Prototyper project. In the absence of the eventual users, the group acted as expert users for the requirements definition and evolutionary design of the application.

The Map Simulation provided the user with a scrollable map presented on a repositionable flat panel within the crewstation (Beagley, et. al. 1994). A vehicle icon was used to indicate the vehicle's position and orientation, (figure H.2), in real time. The approach taken for icon control was the 'Wizard of Oz' approach, first reported by Gould in the 1980s (Nickerson, et. al. 1990). The crewstation's map was slaved to a map in the front of the vehicle which was used to manually update the vehicle's position and orientation. This approach provided a reliable simulation of the combined compass & global positioning system (GPS). Map interaction was provided by a trackerball and a single function button. Map scrolling was controlled by clicking the button over the map area. The map would scroll in the direction indicated by the cursor which changed to point towards the centre of the screen as it was moved around the map area. A set of 'software' buttons on the left of the screen allowed the user to centre the map onto the vehicle's icon, place repositionable map markers and hide or show map markings. The application included background event logging which allowed replay of experimental runs and provided performance measures for route navigation (Beagley 1996a).

The Map Simulation provided an effective and robust simulation of a computer based navigation system for the purposes of the vehicle field trial. The field trial focused on a direct comparison between paper and computer based navigation as represented by the Map Simulation. This revealed a significant reduction in navigation errors committed by vehicle commanders using the computer based map (Beagley 1996b). User interview following the field trial highlighted a preference for the computer based tool over the current paper based approach. Subjects were not informed of the trial's approach to vehicle icon positioning and orientation. Informal discussion following each trial run showed that all trial subjects assumed the Map Simulation to be a fully functioning tool based on a GPS system.

Although the Map Simulation fell outside the definition of a DAVE system module, it provided an important contribution to the group's research programme. This is an example of the in-house software development capability provided by the DAVE project extending the capability of the group beyond the project's initial aims of support for information access.
C. Laboratory Map Tool

![Instruction Screen](image)

Figure H.3 The Lab Map tool (Instruction Screen)

The Laboratory Map tool was one of several supplementary applications developed to support the laboratory based experiments of the group. A previous experiment used SuperCard to develop an application which presented the subject with a workload task administered through a panel of interactive touch-screen buttons (Thody 1993). The interactive panel approach provided an engaging workload task with the ability to record the subject’s responses.

Development of the Lab Map tool represented an evolution of the past laboratory applications merged with components of the recently implemented Map Simulation. The role of the Lab Map tool was to provide a workload task to investigate the effect of reclined posture on subject performance. The investigation was to be undertaken by two external experimenters, with the support of the group.

In contrast to the previous supplementary project’s the developer was not a member of the ‘customer’ group. The value of expert user feedback in the absence of actual users was shown by the Map Simulation development project. Development of the Lab Map tool
used the two experimenters as the source of design feedback in the project's prototype development cycle.

The majority of the application's code and graphics was adapted from the Map Simulation. The resulting design briefly overlaid an observation arc on a map. Military icons were then presented moving across the area of the map in intervals. The user's task was to respond to threats based on set criteria of icon type, position and direction of movement in relation to the observation arc (Newman 1993). Subjects responded by pressing the software buttons modified from the Map Simulation. Response times and errors were automatically recorded by the application. The workload task was controlled by an instruction file referred to by the software at the start of the experimental run. This plain text file could be edited by the experimenters to tell the application which icon to display, where to display it, which direction it should move, etc. This gave the experimenters full control over the final task without the need to learn the SuperCard programming language.

The application proved to be robust allowing the experimenters to complete the full number of planned experimental runs. It met their requirements for a consistent task and automatic gathering of objective data for the comparison of performance. No usability issues were raised indicating that the iterative design of the prototype development cycle was effective in revealing all major design problems.
References


Appendix 1

SuperCard Code Examples

The following code is an example of a standard routine contained by each of the transparent buttons overlaid on the interface's background graphic. This code responds to user interaction by initiating a routine located in the project's script. The code is designed to allow the button to be positioned anywhere on the screen, at any size. It sends a message to the program's log routine and includes the standard help routine.

Code for button - "Example"

```plaintext
on mousedown
    send mousestilldown to bg btn "Example"
    -- Responds once in the case of multiple clicks of the mouse on the button area
end mousedown

on mousestilldown
    global var1, logvar
    -- set up global variables
    set the rect of cd grc "down" to the rect of the target
    -- Display a resizable rectangular graphic over the target button to
    -- provide the impression of button movement
    play "click"
    -- audible feedback
    if var1=="no" then
        DoRoutineA
        put "yes" into var1
    else
        DoRoutineB
        put "no" into var1
    end if
    -- complete a routine depending on the setting of the variables
    set the rect of cd grc "down" to -10,-10,-5,-5
    -- Hide the resizable graphic (after the delay of completing the set routine)
    put "Example" into logvar
    logit
    -- Add the text in logvar plus a time code to the program's log file
end mousestilldown

on mouseenter
    global helpv
    if helpv=="yes" then
        show cd grc "ExampleHelp"
    end if
    -- If help is enabled show the relevant message when the mouse enters
end mouseenter

on mouseleave
    hide cd grc "ExampleHelp"
    -- Hide the help message if displayed
end mouseleave
```
Appendix J - Related Papers


**Designing an Information System for Experts in Vehicle Ergonomics**

N.I. Beagley, R.A. Haslam & K.C. Parsons

Department of Human Sciences, University of Technology, Loughborough, Leics. LE11 3TU

This paper outlines the steps taken in laying down the form of an information system for use by experts in the field of ergonomics specifically in the subject area of vehicle design. The system is under construction for the Vehicle Design and Systems section of the Army Personnel Research Establishment. The approach has been centred on the section's requirements, with the system developer integrating with the section's members as a means of understanding their work practices and information requirements. The proposed environment for development of the system is a Macintosh based "Hyper-media".

**INTRODUCTION**

In designing a computer system the developer is faced with many choices concerning which techniques, tools, guidelines, etc. might be usefully employed. The approach decided upon may dramatically effect the eventual shape of the product and more importantly its success. Studies of systems implemented within organisations have shown a high incidence of failure (Eason 1987). It has been considered that in many cases the failure of these systems could have been avoided through fuller consideration of user requirements (Haslam 1990). Many of these examples have followed traditional development procedures with external developers constructing the system to a rigid specification. Whilst this in itself does not preordain system failure, this approach could lead to distancing
between the developer and the user. It has been noted that on occasion systems developers think that they follow good human factors practices but that they hold a different understanding of these principles to ergonomists (Gould & Lewis 1987). A recent successful system implementation was completed by a human factors expert who remained in close contact with the system’s user group throughout the system’s development (Wadsworth 1990). It is valuable to consider the approaches of previously successful systems in addition the pitfalls demonstrated by unsuccessful systems.

This paper outlines the approach taken in the development of an information system designed to assist a small group of experts in the area of vehicle ergonomics.

The design process has included the following sections.

- Determination of the information requirements of the group.
- Identifying which information could be usefully manipulated by an IT system.
- Devising the optimum method of storing, retrieving and displaying the information.
- Subsequent construction, evaluation and iteration of the system.

USER CENTRED APPROACH

Although widely accepted in the Human Factors community as important for the effective design of systems, a truly user-centred approach is difficult to achieve. In an ideal situation the users might build the systems themselves as they are the best equipped to understand their needs. Even if this were feasible there are benefits from the design approach of a separate developer. The relative isolation of a developer may permit a more impartial and objective assessment of the user’s requirements and limitations. In addition the developer might bring to the organisation specialised experience or the extra manpower required for this type of development.

In the design of the vehicle ergonomics system the developer has integrated with the user group by participation in day to day tasks, along side members of the group. The aim of this was to assist the developer in forming an understanding of their work practices and information requirements. This approach helps to increase the communication between the developer and the users that is central to a true user centred approach.

In parallel with the involvement in group tasks, formal analysis of the working practices of the group has been carried out. A questionnaire was completed by individuals in the group concerning their work practices. In addition, analysis of the time allocation data collected by the organisation, for management processes, over a three year period was considered. The conclusions derived from these studies in conjunction with group discussion help to
provide a direction for the future information system by highlighting the group's major task demands and how these might vary in the future. Design of a computer based information system should have an eye to the future. If the time scale for completion of the system is two years ahead it is important that the system be designed to take into account any change in user requirements where these are possible to predict.

The developer cannot rely solely on focusing on the user group for the system's success. Although this system is to be tailored to a small user group it is important to consider organisational policies such as computer system standardisation.

INFORMATION SELECTION

The determination of what information should be contained within the system has been user driven, based around structured questionnaires, informal discussions and semi-formal group meetings. Informal discussion with individual group members illustrated the general areas which they considered might be benefited by the application of IT. This was expanded and formalised by a structured questionnaire completed by every member of the group. The results of the questionnaire helped to develop topic headings for inclusion in the information system. The headings were then presented to a group meeting for discussion. The product of this group meeting was a ranked list of information subject areas for inclusion in the system. This provided a basis for initial development of the system and a platform for further discussion by the group. This specification process not only produced a list of prioritised topics, but stimulated a consideration, within the group, of what they could get from such a system. By negotiation of the system's content it is hoped that all the members of the group will understand and agree with the compromises necessary for system specification. Subsequent informal conversations have highlighted additional ideas and potential modifications to the formal list.

INCREMENTAL PROTOTYPING

In order to take advantage of the increasing awareness within the group of the system potential, the users must be given the freedom to expand and modify their initial specification for the system. As a result the method of the system's construction must be flexible enough to accommodate these changes.

The most suitable development environment was considered to be a Macintosh based hypermedia. Available on this platform are the two well established hypermedia programming environments HyperCard and SuperCard. These packages incorporate object oriented programming languages, HyperScript/SuperScript, to generate stand-alone applications. They allow rapid development of functional programs which retain the flexibility to permit modification. These packages have been successfully applied in several fields ranging from the simulation of commercial
aeroplane products (Hofer & Ruggiero 1990) to human factors information systems such as Mil.Std.-1472D.

SuperCard has been chosen to develop the vehicle information system. SuperCard is already used within the section for rapid prototyping and as a result the group is generally familiar with the Macintosh environment. Building on the experience already gained using this package for rapid prototyping it will be used as the development tool for the system following an incremental prototyping approach. As such the "final" system will have evolved from the initial prototype. As a prototype, the program should be easy to modify and expand at every stage of the its development.

The acceptability of prototyping for development of the system will be dependent largely on the time taken for it to produce a functional and reliable program. By employing a prototyping approach to design the group can specify the function and appearance of the system throughout its development. This is important as the group members will have a better idea of what they want from the system once they have seen an example of what it can do. By involving the group in evaluation and iteration of the system throughout its development the future system should include what the group wants from it as opposed to what the group predicted they might want from it at the outset.

SYSTEM INTERFACE

Interaction with the system will be via a graphical user interface (GUI). The screen layout and communication protocols will be based on guidelines previously established. Rapid advances in technology have increased the choice of input and output devices available for use with the future system. Initially the system will use mouse, keyboard & scanner input with a 19" colour CRT output. This does not rule out other devices. Touch-screen, speech recognition, handwriting, etc. might be incorporated for input if considered to be of potential benefit. Evaluation and user feedback will indicate whether these devices are suitable for this system. Presently the output is conceived to consist of interactive screens of static information. This could be expanded to include access to CD-ROM archives, video, audio, etc.

EVALUATION

As a consequence of the small group size for which the system is being developed a fuller consideration of changing requirements of the individuals can be included in system development. The small group size does however impose restrictions on the quantity of data which can be gathered in the system's evaluation. Evaluation of the system throughout its development is vital in order for the system to evolve. It is proposed that a combination of objective measures, subjective feedback and expert evaluation might be employed to provide sufficient information for the iteration of the system. Objective measurement will be achieved by monitoring system usage in the background. More detailed information can be expected through informal discussion with the individual users.
though this may be supplemented formally using structured questionnaires.

OVERVIEW

Construction of the first module of the system is underway in the subject area of Anthropometry. The range of data was specified by the group via a questionnaire. Group feedback from the first semi-functional prototype has provided guidance for further development of the prototype in both the areas of functionality and screen design. In this user centred, incremental prototyping approach it is hoped that the information system produced will achieve its objective of improving the group's ease of access to relevant data.

REFERENCES

Eason, K., 1988, Information Technology and Organisational Change (London: Taylor & Francis)


A Computer Based Tool for Accessing Anthropometric Databases

N.I. Beagley, R.A. Haslam & K.C. Parsons

Department of Human Sciences, University of Technology, Loughborough, Leics. LE11 3TU

Based on the requirement of the Vehicle Design and Systems section of the Army Personnel Research Establishment for improved access to their anthropometric database, a computer based tool has been developed to assist data retrieval. The utilisation of the improved graphical capability of desktop computers using a Macintosh based "Hyper-media" environment, presented the potential for novel approaches to the presentation of information, as compared with traditional database query systems. The opportunities and pitfalls of this extended freedom in interface design as encountered in the design of this anthropometry tool are discussed.

INTRODUCTION

In response to the requirement of the Vehicle Design and Systems section of the Army Personnel Research Establishment for enhanced access to the anthropometric data required for their work in the area of Vehicle Ergonomics, a project was initiated to develop a bespoke computer based tool to assist the section in their work. The scope of the project dictated a modular structure for the information system, based on prioritised information categories (Beagley, Haslam, Parsons 1992). This paper discusses the design, development and implementation of the Anthropometry module of the D.A.V.E. system (Database Application in Vehicle Ergonomics).
BACKGROUND

Work on the development of this tool has been undertaken by a single developer working amongst the eventual user group. The approach to the system's development is based upon a user driven evolution of the system through iterative prototyping.

This work has been undertaken using SuperCard on the Macintosh computer. This is a high level development environment which provides a method for non-programmers to create bespoke hypermedia applications. Continuing advancements in computer technology has provided affordable computers, capable of providing the processing levels required to adequately support high level programming code and colour output on a large screen. This opens the door to a large number of potential developers who now have a greater range of choices in the design of their interface.

As a result of technological growth there is an inevitable lag in the standards and guidelines needed to support the decisions these designers must make in the development of their systems. The incremental prototyping approach to in-house system development reinforces system design and provides a method of identifying problems involved in interface design which are, as yet, undocumented.

INFORMATION SELECTION

An important component of system design is to ensure that the information to be contained in the system is actually required by the user. Redundant information inevitably impedes user's access to the information they require.

This system was to be designed primarily for the use of the Vehicle Design Section. As a result it was possible to restrict the spread of data to be included in the initial version, to that which was specifically relevant in the work of the section. It was decided that the primary source of data would be the Anthropometric Survey of the British Army (Gooderson 1982). This however contains measures such as skinfold thickness which were unlikely to be required during the normal course of work of the group. Similarly, surveys of certain populations have more bearing on the work of the group than others. Through the application of a questionnaire to the members of the user group, a list of required measures and populations was compiled. The motivation for determining relevant data was to enhance the future user interface through the removal of redundant information. Whilst the Anthropometry module has been designed around data appropriate to vehicle design, the database itself was left intact to allow expansion of the interface to include omitted measures, should a user requirement for these measures be seen.

SYSTEM INTERFACE

The combination of Hyper-media development software and computer support for large screen (19"+) colour graphics expands the potential of system interface design. Command line interfaces appeared as a consequence of the limitations in the ability of early machines to display
information. In the absence of a true natural language interface the command line interface requirements for a user knowledge of commands and syntax acted as an obstacle to many potential computer users. The move to support full screen interfaces whilst not displacing the command line interface did open the door to the development of graphical interfaces. These interfaces, in the form of the WIMP interface of Windows and the Macintosh Operating System are set to dominate the computer market by opening it up to the large population of potential users. Hardware development has not stopped there. In order to safeguard the usable lifetime of a new system the developer should review hardware trends and consider employing recently released output devices for which guidance in usage is unavailable. The diversity of computer output devices presently available raises numerous questions for the interface design of future computer tools.

The 19" colour monitor output format was chosen for the development of the D.A.V.E. system in preference to a more usual 13" screen due to the potential for presenting, either more information on screen, or the same amount of information more clearly.

The appearance and function of the interface was influenced by the following principles in addition to more specific interface design guidelines.

1. To minimise the depth of the system (i.e. Reduce the number of levels through which the user is required to navigate in order to find the required information)

2. To provide an intuitive interface through the adoption of a 3D metaphor for the appearance and function of the interface components.

3. To provide the user with "on-line" help through demonstration and/or description depending on the user's preference.

4. To tailor the information specifically to the user group, thereby avoiding redundant information where possible.

5. To maintain consistency within the interface in order to provide a blueprint for future modules.

In the design of the Anthropometry module's interface it was decided to use the extra screen space to present more information than would have been legible on a 13" screen. This decision was based on the aim of minimising the system's depth as a trade off against potential screen clutter. By reducing the depth of the system the goal is to simplify navigation within the system and provide rapid access to the relevant information.
It was found that the 5th & 95th percentile measures for populations were most regularly required in the work of the section. Therefore an attempt was made to present these values for all the measures the group specified on the first screen. Consequently the user is only required to interact with the system when they require data different from that normally used. In order to assist the visual search through the 35 measures presented on screen, a pictorial index was devised with which the user can highlight the list dimension by pointing to the measure of interest on a representative photograph. The concept of online help was expanded through the provision of a demonstration option for the user in addition to the usual descriptive paragraph. The help menu is revealed by pressing the help button which will be located in the top right of the screen of all D.A.V.E. modules. The descriptive paragraph is displayed when the user passes the cursor over the function's description in the menu. If the user clicks on the descriptive line of the menu, the actions required to complete the queried function are demonstrated on screen in conjunction with a verbal description of what is being carried out. This mimics the way an experienced user might demonstrate a function to a naive user. The parallel verbal description helps to set an acceptable pace to the demonstration.

Guidelines and standards were unavailable to support all the decisions made in these design features of the interface. As a result, it has been necessary to base any measure of acceptability on the ability of the group to use...
the system. This usability assessment is integral to the
development process due to the incremental prototyping
approach taken.

INCREMENTAL PROTOTYPING

In order to allow maximal user involvement in the
specification and design of the system the approach to
development has been through incremental prototyping. Using
this technique, the designer's early concepts of the system
are quickly cobbled into a semi-functional program which
will eventually evolve into the "final" product. The
designer is able to use an early prototype to test whether
the design is capable of meeting the functional
requirements of the system, as well as conforming to
interface guidelines and standards. At this stage,
evaluation of the prototype by experts in Human Factors can
provide important criticism concerning usability
considerations.

Informal involvement of the members of the user group
throughout the system's development provides them with an
understanding of the abilities and limitations of the
computer system. From this they were able to form a
stronger concept of what they would ideally like the system
to do for them. Incremental prototyping using Hyper-media
provides the flexibility to allow the specification of the
system to evolve along with the developing system.

Using the incremental approach, the limited resources
available for developing the system can be channelled into
development of the system's functionality or interface,
depending on which factors are highlighted in the
evaluation of the previous prototype.

EVALUATION

In addition to the informal evaluation of the evolving
system carried out through user group involvement in the
module's design, the module has been formally evaluated at
two points in its development life, to date. An early
prototype was evaluated by a group of Human Factors experts
who commented on the functionality and interface of the
prototype. Although possessing the same skills as the
developer it was hoped that they could highlight problems
missed by the developer as a result of his close
involvement with the development process. Problems
highlighted at this stage included interface consistency,
measure list grouping, omitted measures, etc. The
criticisms received were either countered by a rationale
for the approach taken or implemented as far as possible
within the development time scale. The iterated prototype
was then enhanced to full functionality for formal
presentation to the user group.

The user group, whilst encouraged to use the system,
were not required to undertake a tutorial in the operation
of the system before its evaluation. Instead, as a method
of introduction and evaluation they were given a task sheet
to be completed in their own time and at their own pace.
The tasks required the users to find numerical values for
anthropometric questions similar to those encountered in
the section's work. The task sheets tested whether each individual was able to make efficient use of the system's functionality with time taken and accuracy as indicators of the usability of the system. The aim of the interface is that it is ultimately to be intuitive. To test this, the evaluation did not follow instruction in the use of the system but rather each individual's approach to retrieving information from a basically unfamiliar interface. In addition to the task sheet results, background logging of the user's actions provided a picture of the different approaches taken by the members of the group to interrogate the system. By highlighting specific difficulties encountered by the users it has been possible isolate usability considerations which may be peculiar to this restricted user group. By tackling these group specific problems it is hoped to ensure the suitability of the system to provide the information required by the section.

CONCLUSION

By taking a user centred approach to development through the technique of iterative prototyping, it has been possible to develop a novel interface for the access of anthropometric data. The success of this technique is dependent on the ability to quickly and easily remodel the program to meet the iterative requirements of the user group.

REFERENCES


Def. Stan. 00-25 (Part 7), 1986, Human Factors For Designers of Equipment: Visual Displays

MIL-STD-1472D, 1989, User-Computer Interface, Chapter 5.15

Maddix, F., 1990, Human-Computer Interaction Theory and Practice (Ellis Horwood)


This work was carried out with the support of the Army Personnel Research Establishment, Ministry of Defence.
Abstract

As a consequence of the availability of high level, software development environments, self tailored software can now be developed by small organisations for whom this form of tool was previously unavailable. As the ability to produce effective software without low level programming language experience is recognised widely, the number and diversity of software applications is likely to greatly increase. By taking an iterative prototyping approach to development an in-house developer can tailor products to meet the actual requirements of the real user group.

1. Introduction

In response to the requirement of a small group working in the area of vehicle ergonomics to access anthropometric data required for their work, the development of a computer based tool was initiated. Work on the development of this tool was to be been undertaken by a single developer working amongst the eventual user group [1].

The risk of failure of systems implemented within organisations is high [2]. It has been considered that in many cases failure of such systems could have been avoided through a fuller consideration of user requirements [3]. In-house development for small user groups can present immediate advantages to a developer, through user contact. Where the success of a system is largely determined by the user group possessing the ability and inclination to use the system, the in-house developer has a head start on remote development teams. Effective use of the advantages offered by continual user contact provides the in-house developer with the potential for ensuring the fit of the system to the user both in terms of the system’s functional requirements and interface preferences of the user group. As a system can only be successful if it meets the actual
requirements of the user group, the danger remains of system failure if this user contact is not adequately utilised

2. Specification

The system must :

1. Meet the information requirements of the user

Following the analysis of the working practice of the group the individual members were asked to consider their information requirements. From this base, the topic heading of Anthropometry was arrived at, through group discussion.

Refining this broad heading into a specification for development requires inputs from many areas; the data available, the limitations of the development platform and not least the limitations of the users.

The starting point for the tool was a comprehensive anthropometric database. This, however contained information irrelevant to the vehicle design work of the group. Redundant information can impede the users’ access to the information they require. This however must be balanced against not providing sufficient information. Due to the specific area of work to be supported by this tool, it was possible to specify a reduced range of anthropometric measures to be presented to the user.

2. Be accessible to every designated user

Despite a range of expertise within the group, the areas of information required were broadly similar. There was however a variation in working practices, as a consequence of age and experience. This was particularly evident in the degree to which each member utilised computers in their work. Generally computer usage within the group was low.

In order to meet the requirement that all designated users be able to access the system it was necessary to produce as simple and intuitive a product as possible, within the specified information domain.

Where possible, the system should provide answers to typical questions. The users should not be required to reformulate the question unnecessarily in order to find the answer to their enquiry. For the anthropometric tool the typical enquiries were considered to be in the forms a) What is the stature of a 95th percentile male, b) Given a ceiling height of x, what percentile male could be accommodated. It would be quite possible for the system to include multiple functions in the form of scatter graphs or correlation tables, etc. but unless the user group specifies a requirement for these functions they would serve only to increase the complexity of the interface.

Having specified the functionality and range of information it is possible to construct a prototype.
3. Evolve with the user's requirements

Although it is important to construct a specification before embarking on the development of a product these requirements should not be cast in stone. Over the course of an in-house development both the users and the developer get a firmer understanding of the group's information requirements in addition to the potential and limitations of the future product. It is important therefore that the product be able to evolve to meet the developing requirements of the users. This flexibility to meet current requirements is provided by the iterative prototyping approach to development. This approach requires a careful choice in development tool as it must allow considerable flexibility in design throughout the system's development and iteration.

3. Development

This work has been undertaken using SuperCard on a Macintosh Quadra 700 computer. SuperCard is a high level development environment which provides a method for non-programmers to create bespoke hypermedia applications. These packages have been successfully applied in several fields ranging from the simulation of commercial aeroplane products [4] to human factors information sources such as Mil.Std.-1472D. Continuing advancements in computer technology have provided affordable computers, capable of providing the processing power required to support high level programming code and large screen, colour output. This opens the door to a large number of potential developers who now have a greater range of choices in the design of their interface.

As a result of technological growth, there is an inevitable lag in the standards and guidelines needed to support the decisions these designers must make in the development of their systems. The incremental prototyping approach to in-house system development reinforces system design and provides a method of identifying problems involved in interface design which are, as yet, undocumented.

The major impediment to an effective incremental prototyping approach is the inertia resulting from the time and cost associated with program modification. The short learning curve of SuperCard as a development medium relative to lower level programming environments allowed prototyping of fully functional models of the system which could be altered as improvements and additions to the system became apparent. As a result of the low time cost of program generation and alteration, it was possible to assess novel interface approaches to the presentation of the data.

Following the hardware improvements which replaced single line textual output with full screen graphical and textual output, the graphical user interface has gained in popularity. In the absence of a true natural language interface the command line interface requirements for a user knowledge of commands and syntax acted as an obstacle to many potential computer users. With the associated reduction in user memory load, graphical interfaces, in the form of the WIMP interface of Windows and the Macintosh Operating System,
dominate the computer market by opening it up to a large population of potential users.

The advantages of a graphical interface can be increased still further by employing colour and a large screen area. A graphical interface approach was adopted for the development of the Anthropometric tool based on a 19" colour monitor. Tailoring the system to the user group includes fitting the software to the equipment the group will have access to. In this case it is possible to ensure that unlimited access to the tool can be provided through large screen colour monitors. As a result it is not necessary to accept the limitations of small monochromatic displays in the design of the interface. Often commercial products must compromise the interface in order to support low end terminals.

The appearance and function of the interface was directed by the following principles in addition to more specific interface design guidelines [5].

1. To minimise the depth of the system (i.e. Reduce the number of levels through which the user is required to navigate in order to find the required information)

2. To provide an intuitive interface through the adoption of a 3D metaphor for the appearance and function of the interface components.

3. To provide the user with "on-line" help through demonstration and/or description depending on the user's preference.

4. To tailor the information specifically to the user group, thereby avoiding redundant information where possible.

5. To maintain consistency within the interface in order to provide a blueprint for future modules.

Navigation through multiple screens in order to find target information takes time and has the potential of disorienting the user. The aim was to provide the information user required by the user with the minimal need for user navigation, by reducing system depth. The final interface design was condensed to a single screen. This was made possible by the removal of redundant information and the use of a large screen area.

The resulting interface displayed all available anthropometric measures in two lists on screen. Beside the measures lists were two dimensions lists which were set up to display the most commonly required data for each measure (5th & 95th percentiles). The users are required to interact with the system only when they require data different from those most frequently used. In order to assist the visual search through the 35 measures presented on screen, a pictorial index was devised. The user can highlight a list dimension by pointing to the measure of interest on a representative photograph beside each list. The survey populations to be included in the results, although pre-set on the most relevant data, can be chosen by clicking on the
check box associated with each measure. The total number of individuals included in the selected population combinations is indicated in a total window. The interface as described is designed to provide an unknown dimension when the users know the measure, percentile, population and units they require.

The alternative enquiry format in which the user knows a dimension, the related measure, population and units but wants to know which percentile of that population would fit can be answered by pressing the find button (bottom left). The user will then be asked for a dimension relating to the selected measure. The system will return the percentile values which fall either side of the specified dimension. A measure is selected, by clicking on its’ name in the list or on the representative photographs. Once selected, the measure’s name and details are displayed in the message box at the bottom of the screen.

Figure 1. Anthropometry module interface.

Help is provided when the user presses the large help button at the top right of the screen. The concept of on-line help was expanded through the provision of a demonstration option for the user in addition to the usual descriptive paragraph. User feedback showed that demonstrative help was popular with the group. Background logging of initial system use showed that half of the group made extensive use of demonstration whereas the remainder primarily
simply referred to the descriptive paragraph. The initial use of a question mark icon for the 'help' button was shown to be confusing and was replaced by the button's name.

The data contained within the screen, as set up by the user, can be printed by pressing the print button (top middle of the screen). Printing was found to cause of problems for the group as the system required them to first select the measures to be printed. As a result the printing option was simplified to print the entire contents of the screen. Finally the user can return to the main entry screen by pressing the button at the top left of the screen.

As the members of the group were largely inexperienced in the use of computers an attempt was made to provide a simple interface which they might find intuitive. This was done by employing a 3D metaphor. The graphical orientation of the SuperCard development environment allows realistic representation of hardware. The top level entry screen has the appearance of a raised metal panel in which are set buttons. These buttons when selected to move the user to the various modules of the system. When a button is clicked on, or pressed using a touch-screen, it provides feedback by appearing to depress and by clicking. A message is then displayed to the users on a simulated LCD panel to notify them that their selection is being loaded. The anthropometry screen retains the hardware appearance in the form of the title bar and the return and help buttons. The 'screen' section of the interface includes 3D cues to function in a more stylised form.

The designer of a system is presented with a large array of techniques, standards, guidelines, etc. to help ensure the suitable design of a system. When considering the interface potential of new hardware, guidance is not always available to the developer. In the case of system development for a small user group, iterative prototyping in a hypermedia environment can quickly test the acceptability of a design, throughout a system's development.

4. Conclusion

At the level of in-house software development for small user groups, a hypermedia environment provides a suitable platform for bespoke tool development, provided a flexible approach is followed, based on the iterative evolution of the system. It is important to exploit the advantages of continual user contact when developing systems for small user groups by ensuring their involvement throughout the project. As a consequence, small organisations who identify potential benefits in a computer based tool, not catered for in the commercial market, should consider this method of software development as a method of acquiring a novel tool.
5. References


2 Eason, K., 1988, Information Technology and Organisational Change (London: Taylor & Francis)


5 Maddix, F., 1990, Human-Computer Interaction Theory and Practice (Ellis Horwood)

This work was carried out with the support of the Army Personnel Research Establishment, Ministry of Defence, U.K.
Designing an Electronic Map Simulation

N.I. Beagley, R. Edwards & D.F. Streets

OPC Section,
Army Personnel Research Establishment,
M.o.D., Farnborough,
Hants. GU14 6TD

One of the most useful tools promised for a future vehicle is the position indicator. When combined with the information content of a map, the task of navigation is greatly simplified. A recent trial in support of future vehicle concepts required the inclusion of an electronic map. The limitations of map systems currently available in terms of cost and flexibility led to the in-house development of an electronic map simulation for use in the field. The limited time and resources for development of the system demanded a resourceful approach. The result was a rugged map system that can be readily evolved towards an increasingly representative tool.

Introduction

The progress of technology has made tools and utilities, unforeseen five or ten years previously, commonplace in everyday life. The consideration of concept systems proposed for development over an extended time scale must include those tools that may become available during the development period. The task of predicting what tools may appear, or in which direction current tools may develop, is not simple. Evaluating the effect of such concept tools is still harder. This paper documents and discusses the construction and implementation of a predicted tool as applied to a future complex system in the form of a concept vehicle.

The tool took the form of a computer based map and global location system to assist in the navigation and basic communication for the vehicle's crew. Global positioning systems (GPS) are currently available based on triangulation from fixed terrestrial beacons or from navigation satellites. Since the introduction of these systems, they have improved in terms of both accuracy and cost, and are continuing to improve in both respects. This has increased the application range of this technology. GPS systems previously restricted to military aircraft are now finding their way into widespread applications from navigation for hill walkers to the tracking of vehicles such as taxi fleets. Systems incorporating map images and positioning systems are becoming available on low cost computer platforms. Although currently unavailable, it was anticipated that further development will lead to a low cost computer based navigation tool incorporating visual map information and live global positioning data.

As a subset of a larger evaluation of concept vehicle's features, it was considered necessary to provide the vehicle's crew members with a representative "electronic map", incorporating global positioning and limited communication as a full replacement to the current method of paper map navigation. This tool was needed in order to study the human factors requirements for future military vehicle based systems. The research group was unable to find a commercially available map system meeting their functional specification. It was decided to provide the test vehicle with a computer
based map simulation which was to be developed in-house by the group. This approach was adopted as a low cost solution, retaining full control over the specification and design of the system within the limits of the available equipment, development tools and invention of the developer.

**Design, Development & Iteration**

The approach to development taken was iterative prototyping in a hypermedia environment on the Macintosh desktop computer. Hypermedia has been employed by the group in the parallel development of the Database Application in Vehicle Ergonomics (D.A.V.E) Beagley, Haslam & Parsons (1992, 1993) This provided the base of developing experience in the use of the SuperCard® development environment. The suitability of hypermedia for the development of the DAVE system gave the confidence in the environment to provide the interface control and background functionality necessary to meet the proposed specification of a computer based system.

Access to the functions of the map, such as scrolling the map and the input of symbols were provided by pressing software buttons displayed on-screen. The SuperCard environment allowed the graphics and control scripts for the buttons to be copied from SuperCard programs previously developed within the group. These imported entities could then be tailored to the specific requirements of the new program.

The initial specification for the map system was loose, providing the core requirements without prescribing the details of how each requirement should be reached. The provision of each function was based on the scope of the development environment in conjunction with human factors guidelines for interface design. Each function was quickly prototyped and tested by the developer in a small iterative cycle to discover and amend any errors in the code. In addition, this stage provided a first line of defence against functional inconsistency and usability problems. This low level iterative cycle forms part of a nested cycle of iteration present throughout the development cycle (figure 1).

![Nested Iterative Prototyping](image)

**Figure 1. Nested Iterative Prototyping**

The flexibility of hypermedia as a prototyping environment allows the incorporation of frequent testing of each aspect of the evolving prototype throughout the development process. At a higher level iterative cycle the functionality and usability of the prototype was assessed by the research group. As the pilot application formed only a small element of the concept vehicle's features being studied, the resources allocated to the applications development were limited. Evaluation using a representative user group was therefore restricted to the first vehicle trial. The nature of the application was an experimental tool to study human factors requirements, as opposed to a system for actual implementation. For this reason a truly, end user centred approach was not required. User requirements and limitations were however considered throughout the development process as the research group, who provided
the expert evaluation, were human factors experts with wide experience of the potential user population.

Pilot Field Trial

The pilot application was based on a Macintosh Ilfx and installed in APRE's mobile reconfigurable crewstation (POD) for a field trial in 1992 Streets, Edwards & Gosling (1993). The crew was provided with a CRT, fitted with a touch screen and a trackball with which to interact with the map system. By including the pilot map system in this trial the aim was to demonstrate the feasibility of using a desktop computer based map simulation for field based research. The particular concern attached to this approach was the reliability of the computer hardware for operation in a moving vehicle. This concern proved unfounded with the computer operating faultlessly throughout the trial. The consistent execution of the pilot program confirmed the suitability of hypermedia as a development environment for this application. Difficulty was encountered in linking the system to a Decca Navigation system as a source of location data. Problems with data transfer and loss of signal prevented the consistent provision of location data to the system. As a consequence it was not possible to draw firm conclusions as to the value of a computer map for in-vehicle navigation.

Software buttons positioned around the edges of the screen's display area were used for control of the map. These could be quickly and easily altered when iterating the system. The system was based on a touch screen, allowing the user to operate it by pressing the buttons on the screen. The lack of tactile feedback could be largely compensated for by using audio feedback in the form of a "click" when a software button was depressed. This feedback was occasionally disguised in the moving vehicle due to increased background noise. This left only the visual feedback of the moving button on screen and any function initiated in the system. In addition the touchscreen used did not allow gloved operation. These limitations were accepted in order to retain the flexibility of control over the system's functionality at the time of design.

Whilst forming a small part of the overall findings of the trial, the pilot map's inclusion pointed to areas of research surrounding the use of an "electronic map" base in the POD vehicle.

The "Electronic Map"

Following on from the initial POD trial it was decided that the use of a computer based map should be investigated in more depth. The design of the subsequent POD trial therefore incorporated a paper map/"electronic map" comparison as one of its areas of investigation.

The experience from the application of the pilot map pointed to several areas in which the map system could be improved. One improvement made possible by the advance of technology was the replacement of the cumbersome CRT screen with an active matrix, colour, flat panel, LCD screen for the presentation of the map. It was decided to use this new technology in order to allow the crew to move and reorientate the map when using the system. Although the resilience of the CRT was proved in the previous trial a further precaution was taken by building a protective windowed box in which the new screen was placed.

The use of the LCD screen removed the touchscreen access allowed in the pilot system. The knock-on effects of this decision in terms of interface design were considerable. In order to maintain usability of the system it was necessary to change the interface to take into account the loss of the touch screen as an input device. The changes necessary were made possible in a restricted time scale, due to the flexibility of hypermedia as a development environment. The decision made in the previous system not to provide the crew with a keyboard was upheld. This left the trackball as the only remaining device available for system interaction. The trackball was retained as the preferred positional device as its use in the previous trial was found to be suitable. One of the trackball's buttons was disabled in order to reduce potential confusion in its operation. The use of software buttons was retained, due to the flexibility of interface and functionality control they afforded. The removal of the touchscreen as the method of interaction with these buttons was recognised in the
design of the iterated interface with the reduction of software buttons from 12 to 4 (figure 2). These four buttons were operated by positioning the cursor over the software button and depressing the trackball's remaining button.

![Figure 2. The "Electronic Map"

The primary guideline for the design of the system was simplicity. The system avoids menu levels removing the requirement to navigate through the system. The full functionality of the system is displayed to the user, on-screen.

An important feature iterated from the pilot application was the method of movement of the viewed map area. The new system provided a smooth scrolling map in eight directions as opposed to the previous map's four directional jump movement. The method of control was changed from four software "direction" buttons positioned around the perimeter of the screen, to a cursor which modified map movement depending on its screen position. As the cursor was moved around the perimeter of the map area it would change between eight arrow shapes, each pointing to the centre of the screen. Using the metaphor of a large map being dragged into the window of the "electronic map", the cursor arrow pointed in the direction in which the map would be scrolled when the trackball's button was depressed. The use of this interface device in place of the perimeter buttons reduced the visual complexity of the map whilst increasing the area available for displaying the map. This method of map movement proved usable to the crew in the subsequent trial.

Simulation

The true advantage of an "electronic map" over a paper map is its potential to display its precise location at any time. The difficulty in adequately integrating a Decca navigation system with the pilot map system demanded that an alternative solution be found. The purpose of a simulation is to match the attributes of the real system within a controlled situation without the difficulties of using or developing a real system. In order for the map simulation to match the specified "real" system, the users had to be shown their position on a map of the area in real time. As the researchers had full control of the routes to be navigated by the vehicle, the position of the test vehicle was known at all times. By communicating this known position from
the front of the test vehicle to the crew in the back, via the "electronic map", the benefit of a global positioning type system without the cost, inaccuracy and variable reliability of a real GPS equipment could be gained.

The SuperCard environment allows inter-application communication and inter-computer communication. The benefit of this facility was built into the map simulation. The ability to move the vehicle icon manually using the keyboard was built into the application. At the same time the state of the map was signalled to any external applications able to receive the data. The state of the map, in turn, could be controlled by external applications. As a result, when two copies of this application were linked, any change in the state of one application would be mirrored in the linked application.

One potential problem of simulation is the subject's awareness that what they are using is not real. This may affect their response to the system. The crew in the POD trial directed the vehicle's driver by radio communication from an isolated crewstation on the back of a vehicle. Running in parallel with the crew's "electronic map" was a copy of the map application running on a laptop computer. The position of the vehicle's icon on the map was controlled by the keyboard from the front of the vehicle and relayed to the crew's map via a cable. During each vehicle's run, the controller of the laptop's map was able to update the crew's map in the exact manner in which an accurate and reliable GPS system would be expected to function. The crew was unaware of how the "electronic map" was updated, leaving them to concentrate on how best to use the information it provided them with. As users of future fully functioning GPS map systems, they will not be required to understand the source of the data. This allowed the research group to provide a simulation of a future system that, to the user, was indistinguishable from a "real" system.

As the group had full control over the design of the "electronic map" its potential as a research tool was exploited further by the incorporation of background logging of all user interaction with the system. This provided a source of objective data for every user operation. This data were automatically gathered by the system requiring no extra work on the part of the research group once the logging code had been built in. When considered in conjunction with the trial videos and user observation data, the background data provided a method for identifying and addressing operational difficulties encountered by the users when operating the "electronic map".

Conclusion

Hypermedia, in conjunction with iterative prototyping provided a valuable method for the simulation of a concept system. The result of the development was a realistic tool incorporating the flexibility necessary to iterate the system quickly as requirements developed. It is anticipated that this simulation tool will be continually evolved for incorporation in future trials of the POD vehicle, based on technological trends and on user requirements determined in previous trials.
References


The views expressed are those of the authors and do not necessarily represent those of APRE, M.o.D. or Her Majesty's Government.

Copyright HMSO 1994
Field Based Prototyping

N.I. Beagley

Department of Human Sciences, Loughborough University of Technology, Loughborough, Leicestershire, England

Introduction

Users require experience of a proposed product to be able to specify accurate and stable requirements (Luqi 1989). Prototyping presents a concept in a tangible form, accessible to both the designer and the user. It serves as a tool to both stimulate ideas and refine them, through iteration, towards a product specification that fits the requirements of the user population. User requirements tend to change over the course of development, overtaking initial design specifications (Spence 1991). The process of prototype evolution incorporating user involvement allows user requirements to develop in parallel with the prototype (Beagley 1993a).

Prototyping provides an efficient medium for communication between the user and the designer (Benimaoff, 1989). At the early stages the prototype supports distillation and communication of the designer's understanding of the requirements. At the same time the interface provides the user with an impression of the proposed application. A prototype can take many forms ranging from a sketch up to a fully functional simulation. The choice of prototyping approach depends on the product to be prototyped and the aim of the evaluation.

Software prototypes are most commonly developed as a specification tool and then discarded (Hofer 1990, Windsor 1990). The alternative approach of evolutionary prototyping starts with an initial prototype that evolves through the prototyping process into the final product (Hekmatpour 1987, Langen 1989). The use of prototyping tools to develop end products is increasingly evident (Beagley 1992, 1993b, Poulter 1993, Shalit 1990). The large size and reduced speed associated with high level applications has become less critical due to steady improvements in processor speed and storage capacity. These limitations of prototyping tools are balanced by their flexibility and support for rapid development, allowing applications to evolve quickly into final products. The use of
a prototyping tool allows the developer to concentrate on creating solutions as opposed to overcoming programming problems (Jasany 1990).

Prototypes are a useful method of communication within the design group. They display their real value, however, when applied to user testing. Testing, using a representative user population, is central to effective user centred design. Prototypes allow this testing at an early stage of development to avoid costly errors in specification.

By taking the users out of their normal working environment for prototype testing, there is a danger of missing significant usability issues. The user may be intimidated by a laboratory environment, modifying their approach to the system. In addition environmental conditions such as lighting or noise may impact on the overall operability of a system. It is important therefore that where possible the prototype should be tested in an environment matching its anticipated operational environment.

Construction of a prototype requires the developer to make design decisions, decomposing the requirements specification to a set of design components (Tanik 1989). Prototyping supports innovation by providing a safety-net, allowing the designer to test unusual design solutions. Components, proven through prototyping, can be developed further. Failed components can be replaced by alternative solutions in the iterated prototype. The developer must be prepared to discard components of a prototype as part of this process of iteration. The reduced development time associated with high level programming helps to enhance the disposability of prototype code, reducing the resistance to change.

A Computer Map Prototype

This paper considers the iterative development of a computer map prototype. The prototype was designed as a disposable tool to assist in the requirements specification of computer maps for use in a military vehicle. A fully functional computer map prototype was developed to be tested in the field.

The task of navigating a military vehicle on the move cannot be isolated from the environmental influences. Effects such as vehicle motion, noise and operation in confined space and other parallel activities affect the task of navigation. A realistic environment cannot be easily simulated in the laboratory. The environment chosen for evaluating the computer map was the POD vehicle based in the field (figure 1).
Figure 1 The POD Mobile Reconfigurable Crewstation

The POD is a mobile, reconfigurable crewstation purpose built for human factors research. The POD's shell has been modified to include the vision systems present in a military vehicle. The task of navigation is made harder by the restricted view caused by operating with the hatches closed. The POD vehicle, used in the field, provided a highly suitable platform to test the prototype.

The prototyping tool chosen was SuperCard®, a high-level, graphically oriented programming environment. SuperCard® is a variant of the more common HyperCard™ environment. The simplicity and learnability (Nielsen 1991) of these high-level languages made them suitable as prototyping environments by the group's human factors specialists. SuperCard® was chosen in preference to HyperCard™ primarily due to its ability to support a large colour map.

Whilst important issues surround the physical design of the computer map unit, the scope of the simulation was limited to a consideration of the functional interface. Working within financial constraints, the prototype was developed and implemented on a desktop Macintosh computer. Whilst it may be desirable for a future map system to be repositionable or even hand held the available equipment limited the presentation of the prototype to a CRT screen.

First Prototype

The starting point of any prototype is a concept. The decision to develop a computer map prototype was based on the assertion that a computer map would reduce a
vehicle commander's work load. Whilst similar tools are used in some aircraft, the task differs considerably from military land vehicle navigation. Civil land systems available at the time of concept development were generally restricted to centralised tracking of fleet vehicles. The specific tasks of the user population made this tool individual. Consideration of these related systems informed the initial specification of the prototype without rigidly directing its design.

The population addressed by the prototype was the commander of a future reconnaissance vehicle. The group specified the core functions to be included in a computer map. These were considered to be: 1. Colour map; 2. Current vehicle position (overlaid); 3. Scrollable map; 4. Map marking. This specification was intentionally loose, providing a starting point to the prototype, without constraining creative design. Although it was possible for the group to specify their concept of a future map in greater detail this was avoided to allow the prototyping process to mould the direction of prototype evolution.

Prototype Development

The group had previously used multimedia techniques for the evaluation of hardware panel design. Realistic buttons, switches, lights, etc. were incorporated on a textured background to produce a photographic representation of a panel, on screen. Animation and sound were used to provide feedback when the user interacted with the panel controls. This effect was further enhanced by the incorporation of a touchscreen for panel interaction. The advantage of this approach was that the panel could be quickly reconfigured, retaining full functionality. This 3D approach was incorporated in the design of the computer map prototype. Control of the prototype was by 3D-effect software buttons, simulating hardware buttons. The group proposed that a real system should be controlled by hardware buttons. The flexibility of reconfigurable software buttons supported the flexibility of the prototype in support of iteration.

A 1:50000 colour map was presented at the screen resolution of 72 dpi. The map controls were provided on software buttons surrounding a central map window (figure 2). Size and labelling of the buttons was dictated in reference documents. The act of producing a functional prototype offered the means to tackle problems not covered in the literature, e.g. control of map scrolling in a vehicle environment. The prototype's solution was to place a large button at each side of the screen. Each button's position indicated the area to be revealed, e.g. pressing the top button moved the view to the area hidden above the displayed portion. With the absence of a prescribed solution to this, the act of proposing a solution in the form of a design component provided a starting point for evaluation.
Standard military symbols were used to mark the map. Selection of the map marking icons was controlled by the software buttons. A trackball was used to position the cursor over the map as opposed to using the touchscreen. Touching the screen in any area other than the software buttons had no effect. This was done to reinforce the impression of the software buttons representing their hardware equivalent.

Figure 2. Initial Computer Map Prototype

First Trial

The first prototype user trial was conducted as part of the 1992 POD Field Trial (Streets 1993). Representative reconnaissance crews were used for the trial. This served as a pilot trial to demonstrate the use of the equipment in the field. The desktop computer proved sufficiently rugged to withstand operation in a moving vehicle. The precautions taken included anti-vibration mountings and the prevention of hard-disk access on the move. The screen survived being used by the subjects as a rest for their coffee cups. The plan to provide vehicle location by Decca Nav. triangulation for real-time positioning was abandoned due to insufficient accuracy. The computer map was simply used as a
replacement to the paper map, providing no extra functionality. In this capacity it demonstrated the ability of the crew to interact with the system in a moving vehicle. In addition it highlighted interface issues to be addressed in the prototype's iteration.

**Prototype Iteration**

Information gathered from the prototype's evaluation pointed towards areas that needed improvement. In addition, enhanced equipment became available between the date of the first field trial and the proposed date for the second.

The users occasionally displayed confusion when attempting to scroll the map. The iteration therefore took an alternative approach to the control of map scrolling. Smooth scrolling was implemented, tied to the metaphor of a large map being dragged into the screen's "window" (Beagley 1994). The surrounding buttons used to control map scrolling was replaced by an active cursor. The cursor was displayed as an arrow that changed, depending on its position, to point towards the centre of the active map area. The arrow indicated the direction in which the large map would be "dragged" when the trackball's button was pressed.

The ten buttons used to control the first prototype were reduced to four by limiting map marking to a generic target and a route marker. The functions of centring the map to the non-static vehicle icon and map decluttering were allocated to the two remaining buttons.

The availability of colour flat panels allowed the prototype to be presented on a less cumbersome, repositionable screen (figure 3). The transfer to a flat panel produced knock-on effects such as the removal of touchscreen interaction. This left the trackball and button as the only method of user control. The use of a flat panel, in turn, improved the range of acceptable viewing angle.
The first prototype demonstrated the principle of presenting a paper map on a computer screen. The function that separates it from paper, however, is the ability to overlay the vehicle's current position. Observation of crews using paper maps showed that the majority represent vehicle orientation by turning the map to show the vehicle heading at the top, whilst marking their perceived position with their finger. There were two possible solutions to getting this information onto the map prototype. Firstly, real systems such as a Global Positioning System (GPS), inertial navigation, and an electronic compass could be integrated with the prototype. Alternatively, the information could be simulated using a Wizard of Oz technique (Wilson 1988). Feeding the vehicle's position and orientation in real time from the front of the vehicle provided a functional, reliable, low cost system that was indistinguishable to the users from a real system.
Second Trial

The iterated prototype was trialed as part of the 1993 POD Field Trial. The revised prototype was fully functional and operationally reliable. The trial included a direct comparison between paper map navigation and navigation using the computer map prototype. The subjects consisted of representative reconnaissance vehicle crewmembers.

Measures used include observational data and applied questionnaires. The prototype incorporated background logging of all interaction with the prototype. The time based log could be used to "replay" the session at a later date.

The operational measures including deviation from the route and decision time provided objective support for the benefit of a computer map as a navigational aid. The questionnaire and informal discussion, based on observation, proved to be rich sources of information. It is based on this information that the specification may be evolved further, through iteration of the prototype.

As anticipated the computer map provided each commander with increased confidence in their position. This was due to the added information of position and orientation offered by the computer map. As a consequence more time was spent by the commander surveying the terrain. Due to the simplicity of the design, the crews experienced little difficulty in operating the computer map, despite minimal training. It is important to recognise the importance of simplicity of design in the stressful environment proposed. The overall usability must be considered as the prototype evolves to include increasing functionality.

Although not required for the trial scenarios, a requirement was specified by several subjects for a "zoom-out" function. They considered a reduced scale map representation to be necessary for route planning as it would provide them with an indication of context. This is a good example of the subjects thinking beyond the tasks formally evaluated. They were able to effectively consider how they could use this tool to complete their regular job.

Presentation of the prototype on a flat screen allowed the user to reposition the screen. The user was able to reorient the computer map in the way observed when navigating with a paper map. With one exception, the subjects chose not to reorientate the computer map, despite reorienting their paper map in the comparison scenarios. An important area to be explored in the use of the map is be to compare "north up" map orientation with automatic "vehicle heading up" orientation. A limitation of the prototyping technique at present is the restriction of processing power. Adding the function of map reorientation to the prototype, although possible, stretches the current power of the prototyping tool. The prototype developer must be resourceful in manipulating the prototyping tool to meet unusual specifications.
Future Prototype Evolution

The prototype is a valuable tool away from the field environment. It provides an object for presentation, stimulating debate amongst visiting groups. This debate generates ideas which in turn influence the direction of prototype evolution.

Areas proposed for future development of the prototype map include, integrating a communication link and the use of new technologies such as hand held devices, pen or voice input and remote sensors. The prototyping environments are evolving at a similar rate to the hardware improvements making it possible for the prototype to develop to meet the challenge of evaluating these advances in technology.

Conclusion

Applying a functional prototype in the field, using a representative population allowed the determination of recommendations for in-vehicle presentation of graphical map information. Over the course of development, the prototype evolved from a loose concept into a simulation demonstrating the predicted benefits of the concept tool. The considerations involved in introducing a concept tool into a complex system goes beyond the design of the tool. By using a functional prototype the user trial highlighted knock-on effects resulting from alteration of the overall system.

The strength of prototyping is its flexibility to evolve with the experience of both the designers and users. Creativity in design and disposability in construction supports evolution of design to meet the functional requirements discovered through the iterative prototyping process.

References


Benimoff N. and Whitten W., Human Factors Approaches to Prototyping and Evaluating User Interfaces, AT&T Technical Journal, September/October 1989, 44-55


Jasany C., Develop Application Programs in 120 Days, Automation, February 1990, 34-35


This work was carried out at the Centre for Human Sciences, DRA, Farnborough
(Formerly A.P.R.E.)

The views expressed are those of the author and do not necessarily represent those of Loughborough University, The Centre for Human Sciences, or Her Majesty's Government.
Iterative Development of a Free-Text Retrieval Tool

N. I. Beagley & R. A. Haslam

Human Sciences Department, Loughborough University, Loughborough, Leics., LE11 3TU

Improved scanning and data storage methods have opened the door to document archiving. The value of this method of storage can only be realised when archived material has been effectively retrieved. This paper describes the evaluation of a computer based tool designed to provide a small user group with desktop access to internal reference documents. The tool's interface presents a condensed list of all words contained within the document. Evaluation of the system as part of the iterative development cycle was based on a direct comparison between computer-based and paper-based searching. The results of the evaluation illustrated the separate search strategies adopted for the different media. Whilst overall search performance was similar between the two methods, use of an optimised computer search strategy offers equivalent performance for unfamiliar documents.

Introduction

This paper reports the development and evaluation of a text retrieval module of the Database Application in Vehicle Ergonomics (DAVE). The stated aim of the DAVE system is "To assist the work of a group of experts in the field of vehicle ergonomics" (Beagley et. al. 1992). A modular approach has been taken providing access to discrete sections of data through specifically designed access programs. Previous modules developed for the DAVE system include a module to access the group's Anthropometric database (Beagley et. al. 1993) and a module to formalise the acquisition and display of vision assessment data. The aim of the text retrieval module was to provide on-line access to text resources relevant to the work of the group.

The application of computers to text retrieval has been in many cases based on the card catalogue approach in which information is condensed to accommodate restricted storage space. In these systems a requirement is placed on a person to draw relevant text and keywords from a document to form the basis of the index. This is a large workload overhead that raises the effective cost of maintaining an index. The increased power of personal computers has opened up text retrieval to a lower entry level market (Holloway 1991) There have been significant improvements in the reliability and cost of optical character recognition techniques (Andon 1990). Advances in storage technology have reduced the cost of storing large volumes of data. This opens up an alternative strategy to traditional paper based filing for document archiving. There has been a significant growth in computer based library systems that have access to the entire text of a document (Dunlop & Van Rijjsbergen 1993). A paper document translated to recognised text permits searches based on the entire content of the text. Rather than requiring an individual to undertake the task of distilling keywords from the document, the indexing task can be passed to the computer. Computer systems are now able to effectively manage
the complete text of documents. This allows documents to be added to an archive without the task overhead of indexing. The result however, is a very different approach to searching the document archive. Removing the restriction of searches based on selected keywords, the user is able to search for a word or combinations of words throughout the entire document archive. Dumas (1988) compared searches using controlled language index terms with searches on all terms contained in the full text of document abstracts. The results support a move towards the full text approach to document archiving. Glavash (1994) found that on-line access of the full text of reference documents was preferable to traditional hardcopy retrieval. However, the value of rapid access to the text had to be balanced against the client's requirements for graphics.

The approach taken for the DAVE text retrieval module was to provide the user with the means of accessing the complete text of the chosen reference documents. By producing a complete list of words contained within the retrieved document, the module provides the user with an unrestricted choice of keywords when searching the document. Processing limitations of the desktop computer systems employed are overcome by pre-indexing of the stored documents. When a document is opened, an alphabetical list of all words contained in the document is presented (Figure 1a). Words in this list may be found by visual scanning or by entering a text search string. The word of interest is selected from the list by clicking on it. The computer then presents the user with a list of occurrences of the selected word. The words are presented in context. This context list may be narrowed by boolean filtering for words surrounding the keyword. Where the list contains a small number of occurrences, visual sorting may be more appropriate. Once a suitable word occurrence is found it can be selected by clicking on it. This changes the display from the index screen to the document screen in which the selected occurrence is highlighted as it appears in the reference document (Figure 1b).

The goal of replacing the paper document with computer access was to improve the speed and accuracy of information retrieval. It was initially considered sufficient to simply provide the reference document in the form of indexed text. Early evaluation of the initial text-only prototype highlighted the importance of the graphical figures contained within the chosen reference documents. Consequently the prototype was evolved to include access to photographic representations of all pages contained within the document. Once in the document screen, the user may choose to view a photographic representation of the selected page. The system makes no attempt to index the graphical elements of the document other than to include the textual descriptions accompanying the figures in the recognised text database. The recognised text is arranged in a page format allowing the user to flip through the document page-wise. The module indexes, stores and retrieves text across a network allowing the central maintenance of the document and image store. This also has the consequence of reducing the storage space required on each machine for operating the software.

The approach taken for the development of the DAVE modules has been user centred, iterative prototyping. This approach has evolved to match the characteristics of the development environment (Beagley, et. al. 1993). The development is in-house, facilitated by good lines of communication between developer and the user population. This close association has been used as the basis for the development approach. Luqi (1989) observed that users require some experience of a software product before they are able to produce robust specifications. Graphically based representations of specifications have been proposed as a suitable medium for communicating requirements (Spence 1991). Rapid prototyping goes a stage further by adding functionality to the proposal model. A functional prototype provides a platform for testing program feasibility and assisting the users to determine the system specification.

The software development has been carried out using SuperCard, a high level, object oriented, programming environment. SuperCard's flexibility supports the evolution of the software prototype towards the implemented system.
Aim

The aim of the evaluation was to provide guidance for the further iteration of the electronic file module. The task set for the user group, investigates the ability of the Electronic file module and the paper source to provide the user group with access to vehicle specific reference data. The capability of the module to replace the requirement for the paper source is investigated by a direct comparison of the two media.

Procedure

Organisational constraints limited the time available to the users for the purpose of assessment. It was necessary to minimise the group's time cost for completion of the evaluation task. The target user group consisted of six individuals, each of whom took part in the evaluation. The stated development requirement of tailoring the system to the specific needs of a small specified group dictated the participation of all members of the group in the evaluation task. The task was constructed to allow all stages to be completed within one hour. This estimation was based on the varied experience of the user group in both in terms of computer utilisation and use of the reference document in its paper form.

The subject was seated at a desk in a quiet office. Positioned on the desk was the computer displaying the DAVE entry screen and a complete copy of the Reference Document. The experimenter demonstrated the completion of two example questions. The examples were demonstrated using both the Electronic File module and the paper based Reference document. The subject was then asked to complete the six questions listed on their task sheet, in order of presentation. Each user answered three questions with the aid of the paper document. The other three questions were answered using the Electronic File module. The questions required the subject to write a single line answer followed by the reference page on which they located the answer. The subjects were asked to complete the questions at their own pace with particular emphasis on accuracy as this most closely matched the normal requirements for reference searches within the organisation. The subjects were required to complete the questions in the order of presentation. This order was balanced to overcome the potential learning effect. If assistance was required, the subject was directed towards DAVE's built in help system. Each subject completed the task within office hours.

Measures taken included error rate & completion time. An action log based on observation was compiled by the observer. Valid task actions were pre-defined in a pilot test. Following completion of the task, each subject was interviewed. In each semi-formal interview the user was asked to highlight adverse and favourable features of both functionality and interface with reference to their particular working requirements.

A telephone enquiry log generated by the section was used to identify six categories in which the section has previously received enquiries. The questions were generated by an individual experienced in vehicle ergonomics from outside of the designated user group. The questions were considered to be of equal difficulty by the questioner.

J38
Results

Completion time was measured from the opening of the document to the writing of the answer for the paper based task and from the first key/mouse press to the writing of the answer for the computer based task. Errors were recorded for incorrect answers. Three errors were recorded for answers based on a paper search. Two of the errors related to insufficient detail in the answer. The third related to a complete failure to find the page reference. Two errors were recorded for questions completed by computer searching. Both computer based errors related to insufficient detail in the answer.

Table 1 Completion time (seconds) & Task Actions

<table>
<thead>
<tr>
<th></th>
<th>Average time</th>
<th>Min. time</th>
<th>Max. time</th>
<th>Average actions</th>
<th>Min. actions</th>
<th>Max. actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paper</td>
<td>219.39</td>
<td>77</td>
<td>702</td>
<td>12.22</td>
<td>5</td>
<td>32</td>
</tr>
<tr>
<td>Computer</td>
<td>225.89</td>
<td>52</td>
<td>507</td>
<td>12.39</td>
<td>5</td>
<td>33</td>
</tr>
</tbody>
</table>

User stated problems

- Would like to see more text in the context window.
- Trouble recognising whether the document was in text or photo mode.
- Confused by the movement between the index and document screens.
- Would like to be able to flick between photos quickly.
- Would like the text to include cross-references to related texts.

Observed problems

- Use of Index scrolling in place of faster index search function.
- Occasional visual searching in place of faster and/not filtering.
- Filtering of complete words rather than a wider, partial word, filter.
- Unsuccessful use of document page flipping, mimicking skimming.
- Non-computer user had initial confusion with scroll bars.
- Failure to recognise the full functionality of scroll bars.
- Double click to select a line when single was sufficient.

Discussion

Observation of the task showed two distinct search strategies. The computer tool demands a structured approach in which a successful search is dependent on the distillation of appropriate keywords. Success using the paper source involved a high proportion of skimming through the document. This approach hinged on the individual's previous experience of using the reference document. Comparison of the two media in this evaluation is immediately biased in favour of the paper medium. All of the subjects required the reference document for their work and therefore had experience in searching the paper document for relevant references. It is important to note here that the enhanced ability of individuals to skim through the document was not evident with subjects who had not used the document recently despite considerable previous experience. The subjects had no previous experience of free text searching. Prior training in the use of the computer tool was minimal, as it was designed to support occasional use. The tool must present an intuitive interface to be effective. The evaluation aimed to highlight features of the system which were not obvious to a naive user.

Completion time and error rate provided the best performance measures. The subjects showed a wide spread of performance for both media, demonstrating the loose nature of the task for which the computer tool was designed. The average completion times (Table 1) show a slightly lower average completion time for the paper source. The action list used to record user strategy was based on observable actions. Whilst showing the degree of task complexity, it does not provide a clear measure of performance. The action list serves rather as a record of the adopted procedure. Task analysis based on the
action list showed the deviation of the subjects from the optimal computer approach. Training of the subjects in parallel with functional and interface iterations highlighted in this evaluation provide scope for improvement in the access times for all subjects tested.

Development guidelines dictated the inclusion of a help system for users. Despite the novice level of the users for this evaluation the help feature remained unused. Consideration of the task actions carried out by each subject showed that subjects used the functions of the system appropriately, demonstrating an understanding of the interface. Slow computer search times generally resulted from the failure to choose appropriate keywords. Guidance is therefore required in developing the user's strategy when using the system as opposed to help in accessing the functions of the system which caused few problems for the users.

The strategy of page navigation was used by one subject. This related most closely to the page skimming approach favoured by all subjects when using the paper source. Failure to find the reference using the document's index resulted in skimming. It was possible to quickly flick through pages in the text mode. The picture mode, however, had a system delay of 3.9 seconds making picture skimming unfeasible. When skimming, using the paper source, the subject derived a fast impression of context through diagram recognition. This relates more closely to picture skimming in the computer tool. Effective application of the skimming strategy for the computer tool would require a solution to the update delay encountered when picture skimming. In the meantime the optimal search strategy when using the computer module remains text based filtering.

The computer based document was split into chapters to narrow search areas. The selection of a chapter required a computer processing time of 11.6 seconds. When added to the time required by each subject to select from the chapter list, this delay represented an important proportion of the access times. A recommendation for the iteration of the module was to remove the choice of chapters providing the complete document. In addition to speeding up the direct access to the document's text, this removes the necessity for the user to judge the content of the separate chapter headings based on their titles alone. Removing chapter divisions allows the system to retrieve related references that may be present in other chapters of the document, further increasing the depth of the search.

Providing the reference module on the desktop computer of each user removes the need to overcome the inertia of the user. Using the reference document requires the user to move from their seat, in most cases to another room. When using a computer system an individual makes inherent cost/benefit assessments that dictate the extent to which the functionality of a system is used (Eason 1984). The decision whether to move from one's seat or to choose the computer tool is an extension of this assessment. If desktop access to reference documents reduces the "costs" to the user of accessing the reference document it is expected that use of the reference document will increase.

Iteration Recommendations

- Increase the effective viewing area of the context list
- Improve text/photo mode indication
- Improve index/document mode indication
- Educate the users in the optimal search strategy
- Broaden the document database

Conclusion

The computer generally matched the paper source for the speed and accuracy of specific reference location when used by novice subjects. Whilst matching the access rate of paper, the computer source is unlikely to enhance the access rate for individuals intimately familiar with the paper source. The computer does however offer enhanced access to users less familiar with the paper source. The adoption of a suitable strategy should improve the access times for the user population. As the user group gain experience in applying the computer tool, improved access time for all documents is anticipated. The value of the computer tool to the user group should increase with the growth of the tool's document database allowing users to maintain a high level of searching performance between unfamiliar documents.
References


Glavash, K., 1994, Full-Text retrieval for document delivery - A viable option?, *Online*, May, 81-84


The views expressed are those of the authors and do not necessarily represent those of CHS, DRA or Her Majesty's Government.

Copyright HMSO 1995