Bad Behaviour

The Prevention of Usability Problems using GSE Models

by
Alistair Crawford
B.Ed. Hons. (Griffith University),
Grad.Dip.IT. (Queensland University of Technology),
M.IT. (Queensland University of Technology).

School of Computer and Information Technology
Faculty of Engineering and Information Technology
Griffith University
Australia

Thesis submitted in fulfilment of the requirement for the degree of Doctor of Philosophy.
2005
Abstract

The aim of Human Computer Interaction or HCI is to both understand and improve the quality of the users' experience with the systems and technology they interact with. Recent HCI research requirements have stated a need for a unified predictive approach to system design that consolidates system engineering, cognitive modelling, and design principles into a single 'total system approach.' At present, few methods seek to integrate all three of these aspects into a single method and of those that do many are extensions to existing engineering techniques. This thesis, however proposes a new behaviour based approach designed to identify usability problems early in the design process before testing the system with actual users.

In order to address the research requirements, this model uses a new design notation called Genetic Software Engineering (GSE) in conjunction with aspects of a cognitive modelling technique called NGOMSL (Natural GOMS Language) as the basis for this approach. GSE's behaviour tree notation, and NGOMSL's goal orientated format are integrated using a set of simple conversion rules defined in this study. Several well established design principles, believed to contribute to the eventual usability of a product, are then modelled in GSE.

This thesis addresses the design of simple interfaces and the design of complex ubiquitous technology. The new GSE approach is used to model and predict usability problems in an extensive range of tasks from programming a VCR to making a video recording on a modern mobile phone. The validity of these findings is tested against actual user tests on the same tasks and devices to demonstrate the effectiveness of the GSE approach.

Ultimately, the aim of the study is to demonstrate the effectiveness of the new cognitive and engineering based approach at predicting usability problems based on tangible representations of established design principles. This both fulfils the HCI research requirements for a 'total system approach' and establishes a new and novel approach to user interface and system design.
Statement of Authorship

I declare that this work has not previously been submitted for a degree or diploma in any university. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made in the thesis itself.

Signed

Date.
Table of Contents

CHAPTER 1 INTRODUCTION TO THE STUDY ........................................................................ 1
1.1 The usability problem ............................................................................................... 1
1.2 Pre-deployment and post deployment usability ..................................................... 2
1.3 Approaches to usability analysis and design ......................................................... 4
1.4 Making Usability and HCI more attractive ............................................................ 7
1.5 A Predictive Approach to Usability ....................................................................... 10
   1.5.1 Genetic Software Engineering (GSE) ............................................................. 10
   1.5.2 The Natural GOMS Language (NGOMSL) .................................................. 10
   1.5.3 Comparing NGOMSL and GSE ................................................................. 11
   1.5.4 Justifying an NGOMSL/GSE approach ....................................................... 13
1.6 Contributions of the thesis .................................................................................... 15
   1.6.1 Aim of the research ...................................................................................... 15
1.7 Description of Chapters ......................................................................................... 16
1.8 Summary .............................................................................................................. 18

CHAPTER 2 REVIEW OF THE LITERATURE: USABILITY AND ITS PROPERTIES ........................................................................................................ 19
2.1 Usability design and quality .................................................................................. 19
   2.1.1 The value of HCI and usability ................................................................... 19
2.2 Key usability concepts ........................................................................................... 21
   2.2.1 The user interface ....................................................................................... 21
   2.2.2 Recent interface developments .................................................................... 22
2.3 Understanding and modelling user behaviour ..................................................... 23
   2.3.1 Mental and user models .............................................................................. 24
   2.3.2 Human diversity .......................................................................................... 26
2.4 Usability and related definitions ........................................................................... 28
   2.4.1 Usability engineering, design and objectives .............................................. 29
   2.4.2 Usability testing, inspection and evaluation .............................................. 30
2.5 Usability engineering techniques .......................................................................... 32
   2.5.1 Guidelines and heuristics .......................................................................... 32
   2.5.2 Cognitive walkthrough ............................................................................. 35
   2.5.3 Choosing techniques .................................................................................. 36
2.6 Properties, attributes, principles and quality ....................................................... 37
   2.6.1 Quality and product quality ....................................................................... 37
   2.6.2 Linking quality to quality attributes ......................................................... 40
2.7 Usability and its attributes .................................................................................... 40
   2.7.1 Attributes for error handling ...................................................................... 43
   2.7.2 Attributes for interaction and control ......................................................... 44
   2.7.3 Attributes related to learning and understanding ....................................... 47
   2.7.4 Attributes for user diversity ....................................................................... 51
   2.7.5 User dependent attributes ........................................................................ 53
   2.7.6 Attributes not covered in the study ........................................................... 54
2.7 Summary .............................................................................................................. 54

CHAPTER 3 REVIEW OF THE LITERATURE: TASK ANALYSIS, COGNITIVE ENGINEERING AND SOFTWARE ENGINEERING .................................................................................................. 55
3.1 The freedom of design ......................................................................................... 55
3.2 Task analysis and the early stages of design ....................................................... 56
CHAPTER 6 MOBILE PHONE STUDY: TASK MODELS ......................................................... 247
6.1 Introduction ......................................................................................................................... 247
6.2 The Mobile Phone NGOMSL Model ..................................................................................... 248
   Step A: Choose the top-level user’s goals .............................................................................. 249
   6.2.1 NGOMSL Step B – The Recursive Procedure ............................................................... 252
   6.2.2 Step C: Document and Check the Analysis .................................................................... 253
   6.2.3 Step D: Check Sensitivity to Judgement Calls and Assumptions .................................... 253
   6.2.4 The finalised NGOMSL Mobile Phone model ................................................................. 254
6.3 Building the NEC GSE Model ............................................................................................... 255
   6.3.1 Task 1: Turning the phone on ....................................................................................... 259
   6.3.2 Task 2: Making a phone call ......................................................................................... 261
   6.3.3 Task 3: Add following names and number to phone book (contacts) ......................... 264
   6.3.4 Task 4: Make call ........................................................................................................... 269
   6.3.5 Task 5: Access 3 Web Services ..................................................................................... 272
   6.3.6 Task 6: Navigate Web services to find Java game: Play Golf and download it .......... 273
   6.3.7 Task 7: Take 12 sec video sequence and cancel save .................................................... 276
   6.3.8 Task 8: Use calculator add (5 + 2) ............................................................................... 277
   6.3.9 Task 9: Take photo of glass and save .......................................................................... 281
   6.3.10 Task 10: Create appointment in calendar ................................................................. 283
   6.3.11 GSE Task 11: Create and send email ........................................................................ 286
   6.3.12 Task 12: Create SMS message with “Hello” - Save it to memory ............................... 290
   6.3.13 Task 13: Delete appointment in calendar .................................................................... 293
   6.3.14 Task 14: Send stored photo to 0412 3848 10 ............................................................... 295
   6.3.15 Task 15: Activate java game: Begin game: Close java game ..................................... 298
   6.3.16 Task 16: Delete from phonebook .................................................................................. 300
   6.3.17 Task 17: Delete stored photo ...................................................................................... 303
   6.3.18 Task 18: Delete SMS message ................................................................................... 303
   6.3.19 Task 19: Turn phone off ............................................................................................. 303
   6.3.20 NEC Phone Summary ................................................................................................ 304
6.4 The Samsung GSE Model ................................................................................................. 305
   6.4.1 Tasks in User Tests ....................................................................................................... 306
   6.4.2 Task 1: Turn phone on ................................................................................................. 307
   6.4.3 Task 2: Make call to 0407 515 530 ................................................................................. 308
   6.4.4 Task 3: Add following names and number to phone book (contacts) ....................... 311
   6.4.5 Task 4: Activate call history – find and return call using call history ....................... 313
   6.4.6 Task 5: Accessing the Web Browser .......................................................................... 314
   6.4.7 Task 6: Use Speed Dial ............................................................................................... 317
   6.4.8 Task 7: Create SMS: Save message ........................................................................... 319
   6.4.9 Task 8: Use calculator add (5 + 2) ............................................................................. 321
   6.4.10 Task 9: Assign voice recording to contact ................................................................. 324
   6.4.11 Task 10: Create appointments in Scheduler ............................................................... 327
   6.4.12 Task 11: Use Voice Dial to call (in Figure 6-77) .......................................................... 330
   6.4.13 Task 12: Delete appointment in Scheduler ................................................................. 332
   6.4.14 Task 13: Send saved SMS ......................................................................................... 332
   6.4.15 Task 14: Delete contact from phonebook .................................................................. 334
   6.4.16 Task 15: Turn phone off ............................................................................................ 334
6.5 Summary of the Findings ................................................................................................. 335
   6.5.1 Samsung ..................................................................................................................... 335
   6.5.2 NEC ............................................................................................................................ 337
CHAPTER 7 MAIN STUDY: TEST RESULTS .................................................. 339
7.1 Introduction to the Main Study ......................................................... 339
7.2 Test Procedure ........................................................................... 340
7.3 Usability problems for the NEC tasks ........................................... 344
  7.3.1 NEC Task 1: Turning the Phone On ......................................... 344
  7.3.2 NEC Task 2: Make call to 0407 515 530 ................................. 345
  7.3.3 NEC Task 3: Add to Contacts ................................................. 346
       Comparison to GSE findings ......................................................... 347
  7.3.4 NEC Task 4: Activate Call history – find 0407 515 530 and return call using call history ................................................................. 347
  7.3.5 NEC Task 5: Accessing ‘3’ Web Services .................................... 348
  7.3.6 NEC Task 6: Navigate Web services to find Java game: “PlayGolf” and download it ................................................................. 349
  7.3.7 NEC Task 7: Take 12 sec Video Sequence – Cancel Save ............. 350
  7.3.8 NEC Task 8: Use calculator to add (5 + 2) ............................... 351
  7.3.9 NEC Task 9: Take photo of glass and save ............................... 351
  7.3.10 NEC Task 10: Create lunch appointment in scheduler for October 19 2004, 2 30 p.m. ................................................................. 352
  7.3.11 NEC Task 11: Compose and Send Email .............................. 353
       Comparison with GSE analysis .................................................... 353
  7.3.12 NEC Task 12: Create SMS message with “Hello” – Save to Memory 354
  7.3.13 NEC Task 13: Delete appointment in calendar for October 19 2004:
       230 p.m. (Fixed) .................................................................... 354
  7.3.14 NEC Task 14: Send stored photo to 0412 3848 10: ................. 355
  7.3.15 NEC Task 15: Activate Java Game: Begin game: PlayGolf: Close Java Game ................................................................. 356
  7.3.16 NEC Task 16: Delete all “AI”s from phone Book (contacts) ......... 356
  7.3.17 NEC Task 17: Delete stored Photo ........................................... 357
  7.3.18 NEC Task 18: Delete SMS Message ....................................... 357
  7.3.19 NEC Task 19: Turn Phone off ................................................. 357
7.4 Usability problems for the Samsung tasks ...................................... 358
  7.4.1 Samsung Task 1: Turning the phone on .................................... 358
  7.4.2 Samsung Task 2: Make call to 0407 515 530 .......................... 359
  7.4.3 Samsung Task 3: Add following names and number to phone book ... 359
  7.4.4 Samsung Task 4: Activate Call history .................................... 360
  7.4.5 Samsung Task 5: Access Web Browser .................................... 360
  7.4.6 Samsung Task 6: Use Speed Dial – AI (0407 515 530) ............... 360
       Comparison with GSE analysis .................................................... 361
  7.4.7 Samsung Task 8: Use calculator add (5 + 2) ............................ 361
  7.4.8 Samsung Task 9: Assign Voice Recording to AI (phone book record) 362
  7.4.9 Samsung Task 10: Create Lunch Appointment in Scheduler for October 19 2004 2 30 p.m ................................................................. 363
  7.4.10 Samsung Task 11: Use Voice Dial to call – [AI] .................... 364
  7.4.11 Samsung Task 12: Delete appointment in Scheduler for October 19
       2004 2 30 p.m ............................................................................ 364
  7.4.12 Samsung Task 13 Send saved SMS “hello” to AI .................... 365
  7.4.13 Samsung Task 14 Delete Al from phone Book (contents) ......... 366
  7.4.14 Samsung Task 15: Turn Phone off .......................................... 366
7.5 Overall Analysis ......................................................................... 366
7.6 Summary ................................................................................... 372
Table of Figures

Figure 1-1: Creating the User Experience .......................................................... 3
Figure 2-1: Proposed Attributes Contributing to Usability .................................. 54
Figure 3-1: The Relationship of the NITRD Models in the Early Design Process ...... 55
Figure 3-2: Hierarchical Task Analysis ......................................................... 66
Figure 3-3: GSE Node .................................................................................. 101
Figure 3-4: Primary GSE Objects .................................................................... 101
Figure 4-1: The Key Behaviour Tree Elements ............................................... 112
Figure 4-2: GSE Requirement Example ......................................................... 113
Figure 4-3: Usability Attributes ....................................................................... 115
Figure 4-4: Forgiveness problem 1 – unchecked data ..................................... 119
Figure 4-5: Forgiveness problem 2 – No warning for irreversible action .......... 120
Figure 4-6: Forgiveness problem 3 – Allow preventable error ....................... 120
Figure 4-7: Forgiveness problem 4 – Bad default Value .................................. 121
Figure 4-8: Error Recovery problem 1 in GSE ............................................... 121
Figure 4-9: GSE representation of Error Recovery problem 2 ....................... 122
Figure 4-10: Responsiveness problem 1 ......................................................... 122
Figure 4-11: Responsiveness problem 2 ......................................................... 123
Figure 4-12: Responsiveness Problem 3 ......................................................... 123
Figure 4-13: Responsiveness Possible Solution for Problem 2 ....................... 124
Figure 4-14: Responsiveness Problem 4 ........................................................ 124
Figure 4-15: Responsiveness problem 5 .......................................................... 124
Figure 4-16: Poor Controllability ................................................................... 125
Figure 4-17: Controllability Problem 2 Example ............................................ 125
Figure 4-18: Controllability – Solution to problem 2 ....................................... 126
Figure 4-19: Controllability Problem 4 ............................................................ 126
Figure 4-20: Controllability Problem 3 in GSE .............................................. 126
Figure 4-21: Direct Manipulation Example .................................................... 127
Figure 4-22: Directness problem 1 ................................................................. 128
Figure 4-23: Directness problems 2 and 3 in GSE .......................................... 128
Figure 4-24: Directness problems 4 and 5 in GSE .......................................... 129
Figure 4-25: Directness Problem 6 in GSE ..................................................... 129
Figure 4-26: Directness Problem 7 in GSE ...................................................... 129
Figure 4-27: Efficiency example 1, short cuts for long sequences ................. 130
Figure 4-28: Efficiency example 2, speeding up text entry ............................ 131
Figure 4-29: Efficiency example 3 .................................................................. 131
Figure 4-30: Efficiency Problem 4 in GSE ..................................................... 132
Figure 4-31: Predictability ............................................................................. 133
Figure 4-32: More Predictability Problems .................................................... 133
Figure 4-33: Predictability Problems 4 and 5 ................................................. 134
Figure 4-34: Simplicity example 1 .................................................................. 134
Figure 4-35: Simplicity Example 2 ................................................................. 135
Figure 4-36: Simplicity Example 3 ................................................................. 135
Figure 4-37: Transparency Examples .............................................................. 136
Figure 4-38: Familiarity Example ................................................................... 137
Figure 4-39: Familiarity Problem 2 ................................................................. 137
Figure 4-40: Consistency Problem 1 ............................................................... 138
Figure 4-41: Too Much Consistency - Problem 2 ......................................... 138
Figure 4-42: Configurability Examples ........................................................... 139
Figure 4-43: Low Flexibility ................................................................. 139
Figure 4-44: Flexibility .................................................................. 139
Figure 4-45: A Basic Usability Test .................................................. 141
Figure 4-46: Room and equipment layout for pilot study and case study I ............................................. 149
Figure 4-47: The Panasonic NV-SD20: Does not support G-CODE ........................................................................ 150
Figure 4-48: The Panasonic NV-HD620MK2A: Supports G-CODE ............................................................... 150
Figure 4-49: Main Study Mobile Phones ........................................... 151
Figure 4-50: Triangulating ................................................................. 155
Figure 4-51: Processing raw data into usability problems. ............................................................................. 155
Figure 4-52: Extracting usability problems ................................................................................................. 156
Figure 5-1: Timer Recording Outline in GSE ............................................................................................... 161
Figure 5-2: Overview of Kieras's VCR Model ............................................................................................. 164
Figure 5-3: The Five Steps of the GSE Process ............................................................................................ 165
Figure 5-4: Summary Notation for a standard Button Press ......................................................................... 167
Figure 5-5: GSE representations of NGOMSL assumptions ........................................................................ 168
Figure 5-6: User Goal - Set Clock ................................................................................................................. 170
Figure 5-7: Clock Button Press Event ......................................................................................................... 170
Figure 5-8: Sub goal set day and time ......................................................................................................... 171
Figure 5-9: Set states for day and time ......................................................................................................... 171
Figure 5-10: Clock set state ......................................................................................................................... 171
Figure 5-11: User's Goal has been achieved ............................................................................................... 171
Figure 5-12: Literal Translation .................................................................................................................. 172
Figure 5-13: Set the Clock Method – final variation .................................................................................... 172
Figure 5-14: GSE Representations of Goal 2: Set a Day and Time ............................................................ 174
Figure 5-15: GSE Representation of Goal 3: Set a Value to a Desired Value ............................................. 175
Figure 5-16: Selection Rule Set for Goal Record a Program ....................................................................... 177
Figure 5-17: Selection Rule Set for Goal Record a Program Manually ...................................................... 178
Figure 5-18: Goal 6 .............................................................................. 179
Figure 5-19: Timer Recording in GSE (Compressed vertically to fit on page) ............................................ 180
Figure 5-20: Select Recording Length ..................................................................................................... 181
Figure 5-21: Selection Rule Stop an ongoing recording .......................................................................... 182
Figure 5-22: Stop an ongoing recording .................................................................................................... 183
Figure 5-23: Stop one touch recording ..................................................................................................... 183
Figure 5-24: Method for Goal: Stop a Timer Recording ............................................................................ 184
Figure 5-25: Method for Goal: Cancel a Timer Recording .......................................................................... 185
Figure 5-26: GSE Presentation in this thesis (large arrows show direction only) ................................ 188
Figure 5-27: VCR - Timer Record Layout .................................................................................................. 189
Figure 5-28: VCR --Timer Record Layout ................................................................................................. 190
Figure 5-29: VCR --Timer Record Layout ................................................................................................. 191
Figure 5-30: Timer Programming with the VCR Panel ............................................................................... 193
Figure 5-31: Timer Recording: Set Item Part 1: VCR Panel ...................................................................... 196
Figure 5-32: Timer Recording: Set Item Part 2: VCR Panel ...................................................................... 199
Figure 5-33: Timer Recording: Pick a Value from Values ......................................................................... 201
Figure 5-34: Checking a Timer Program with Remote .............................................................................. 203
Figure 5-35: Setting Timer Programs with the Remote Control ................................................................. 205
Figure 5-36: Entering Programme Position with Remote ............................................................................ 206
Figure 5-37: Setting the remote to Scanner Mode ...................................................................................... 207
Figure 5-38: Tracing Values ....................................................................................................................... 209
Figure 5-39: Setting Items with the remote ............................................................................................... 210
Figure 5-40: The NV-HD620mk2a remote control button layout ............................................................... 212
<table>
<thead>
<tr>
<th>Figure 5-41: HD620 – Timer Programme Screen</th>
<th>........................................</th>
<th>213</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 5-42: Accessing the Timer Programme Screen</td>
<td>.....................................</td>
<td>213</td>
</tr>
<tr>
<td>Figure 5-43: Entering a Timer Recording Part 1</td>
<td>.....................................</td>
<td>214</td>
</tr>
<tr>
<td>Figure 5-44: OSD Timer Recording Part 2</td>
<td>........................................</td>
<td>216</td>
</tr>
<tr>
<td>Figure 5-45: HD620 – G-CODE Programming screen</td>
<td>.......................................</td>
<td>217</td>
</tr>
<tr>
<td>Figure 5-46: Timer Recording with G-Codes Part 1</td>
<td>.......................................</td>
<td>218</td>
</tr>
<tr>
<td>Figure 5-47: Timer Recording with G-Codes Part 2</td>
<td>.......................................</td>
<td>219</td>
</tr>
<tr>
<td>Figure 5-48: Figure G-Code problem – Incorrect Code</td>
<td>........................................</td>
<td>226</td>
</tr>
<tr>
<td>Figure 5-49: G-Code problem 2</td>
<td>........................................</td>
<td>228</td>
</tr>
<tr>
<td>Figure 5-50: VCR Pilot Study – Frequency of Attributes for Examiners 1,2 and 3</td>
<td>.........................................</td>
<td>232</td>
</tr>
<tr>
<td>Figure 5-51: OSD instructions</td>
<td>........................................</td>
<td>234</td>
</tr>
<tr>
<td>Figure 5-52: OSD instructions</td>
<td>........................................</td>
<td>236</td>
</tr>
<tr>
<td>Figure 5-53: VCR Panel instructions</td>
<td>........................................</td>
<td>237</td>
</tr>
<tr>
<td>Figure 5-54: Remote Control instructions</td>
<td>........................................</td>
<td>239</td>
</tr>
<tr>
<td>Figure 5-55: Barcode Scanner instructions</td>
<td>........................................</td>
<td>241</td>
</tr>
<tr>
<td>Figure 6-1: The NGOMSI Process in Full</td>
<td>........................................</td>
<td>248</td>
</tr>
<tr>
<td>Figure 6-2: Physical components of the NEC</td>
<td>........................................</td>
<td>255</td>
</tr>
<tr>
<td>Figure 6-3: NEC system component</td>
<td>........................................</td>
<td>256</td>
</tr>
<tr>
<td>Figure 6-4: The NEC Start Screen</td>
<td>........................................</td>
<td>256</td>
</tr>
<tr>
<td>Figure 6-5: Subcomponents of the Icon Indicator</td>
<td>.......................................</td>
<td>257</td>
</tr>
<tr>
<td>Figure 6-6: Start Screen in GSE Notation</td>
<td>........................................</td>
<td>258</td>
</tr>
<tr>
<td>Figure 6-7: Edit Screen</td>
<td>........................................</td>
<td>258</td>
</tr>
<tr>
<td>Figure 6-8: Turning the Phone On from 3 User Manual</td>
<td>.......................................</td>
<td>259</td>
</tr>
<tr>
<td>Figure 6-9: NEC Task 1: Turn the Phone On</td>
<td>........................................</td>
<td>260</td>
</tr>
<tr>
<td>Figure 6-10: NEC Task 2: Main Steps and screens in making a manual phone Call</td>
<td>..................................</td>
<td>261</td>
</tr>
<tr>
<td>Figure 6-11: NEC Task 3: Making a Call Part 1</td>
<td>.......................................</td>
<td>262</td>
</tr>
<tr>
<td>Figure 6-12: NEC Task 3 Making a Call Part 2</td>
<td>........................................</td>
<td>263</td>
</tr>
<tr>
<td>Figure 6-13: Destination Screen</td>
<td>........................................</td>
<td>265</td>
</tr>
<tr>
<td>Figure 6-14: Name Input Screen</td>
<td>........................................</td>
<td>265</td>
</tr>
<tr>
<td>Figure 6-15: Add Contact Screen for Phone-Contacts</td>
<td>..................................</td>
<td>265</td>
</tr>
<tr>
<td>Figure 6-16: Add Contact Screen for Phone-Contacts Part 1</td>
<td>................................</td>
<td>267</td>
</tr>
<tr>
<td>Figure 6-17: Add Contact to Screen for Phone-Contacts: Part 2</td>
<td>................................</td>
<td>268</td>
</tr>
<tr>
<td>Figure 6-18: Calls Made</td>
<td>........................................</td>
<td>269</td>
</tr>
<tr>
<td>Figure 6-19: GSE Goal 4: Make Call from Call History Part 1</td>
<td>...............................</td>
<td>269</td>
</tr>
<tr>
<td>Figure 6-20: GSE Goal 4: Make Call From Call History Part 2</td>
<td>.........................</td>
<td>271</td>
</tr>
<tr>
<td>Figure 6-21: GSE Goal 4</td>
<td>........................................</td>
<td>272</td>
</tr>
<tr>
<td>Figure 6-22: GSE Goal 5: Connect to Web</td>
<td>........................................</td>
<td>272</td>
</tr>
<tr>
<td>Figure 6-23: Choosing Games</td>
<td>........................................</td>
<td>273</td>
</tr>
<tr>
<td>Figure 6-24: GSE Goal 6: Download Game</td>
<td>........................................</td>
<td>275</td>
</tr>
<tr>
<td>Figure 6-25: GSE Goal 7</td>
<td>........................................</td>
<td>276</td>
</tr>
<tr>
<td>Figure 6-26: GSE Goal 7: Take Video Recording</td>
<td>....................................</td>
<td>277</td>
</tr>
<tr>
<td>Figure 6-27: GSE Goal 8: Part 1 Accessing the Calculator</td>
<td>..................................</td>
<td>278</td>
</tr>
<tr>
<td>Figure 6-28: GSE Goal 8: Using the Calculator</td>
<td>.....................................</td>
<td>279</td>
</tr>
<tr>
<td>Figure 6-29: GSE Goal 8: Part 2: Using the Calculator</td>
<td>...............................</td>
<td>280</td>
</tr>
<tr>
<td>Figure 6-30: GSE Goal 8: Entering an Operator</td>
<td>.................................</td>
<td>281</td>
</tr>
<tr>
<td>Figure 6-31: GSE Goal 9: Taking a Photo</td>
<td>........................................</td>
<td>281</td>
</tr>
<tr>
<td>Figure 6-32: GSE Goal 9: Take Photo and Save</td>
<td>.....................................</td>
<td>282</td>
</tr>
<tr>
<td>Figure 6-33: GSE Goal 10: Create Appointment Part 1 (From the Start Screen)</td>
<td>..................</td>
<td>283</td>
</tr>
<tr>
<td>Figure 6-34: GSE Goal 10: Create Appointment Part2</td>
<td>...............................</td>
<td>285</td>
</tr>
<tr>
<td>Figure 6-35: Manual Reference for Goal 11</td>
<td>........................................</td>
<td>286</td>
</tr>
</tbody>
</table>
List of Tables

Table 2-1: Characteristics of Experts and Novices ................................................................. 26
Table 2-2: Guidelines and Standards ....................................................................................... 33
Table 2-3: The space of interface evaluation methods .............................................................. 36
Table 2-4: Shneiderman's Principles ......................................................................................... 41
Table 2-5: The Ten Usability Heuristics ................................................................................... 41
Table 2-6: User Interface Principles ......................................................................................... 42
Table 3-1: Possible Predictive Properties .................................................................................. 56
Table 3-2 Classifying GOMS techniques .................................................................................. 86
Table 3-3: Task Instances and Task Descriptions .................................................................... 88
Table 3-4: The NGOMSL Process ............................................................................................ 89
Table 4-1: Features of Talk Alouds ........................................................................................... 144
Table 4-2: Subjects for Pilot Study User Test .......................................................................... 145
Table 4-3: Subjects for Main Study User Test ......................................................................... 146
Table 4-4: Tasks associated with timer recording for pilot study ........................................... 151
Table 4-5: Usability Categories adapted from CUE 4-Scenario ............................................... 156
Table 4-6: Severity of Usability Problems ................................................................................ 157
Table 4-7: Comparative summary of pilot study and main study ........................................... 158
Table 5-1: Usability Problems found across the HD-SD20 Task Models ............................... 211
Table 5-2: Problems found on NV-HD620mk2a VCR using GSE .......................................... 220
Table 5-3: Comments for Examiner A ...................................................................................... 224
Table 5-4: Problem Instances 1 – 5 for G-Code Task ............................................................... 225
Table 5-5: Compiled Problems 1 and 2 for G-Code Task ......................................................... 225
Table 5-6: Problems and Attribute Comparison for User Test Task 2 - G-Code .................... 229
Table 5-7: Severity of usability problems .................................................................................. 230
Table 5-8: VCR Study Usability Problem Distribution .............................................................. 231
Table 5-9: Attributes and Codes used in the User Tests ........................................................... 231
Table 5-10 User Feedback by subjects S1 to S4 .................................................................... 233
Table 5-11: The top three problems by severity ...................................................................... 233
Table 5-12: Problems and Attribute Comparison for User Test Task 1 - OSD ...................... 235
Table 5-13: Problems and Attribute Comparison for Task 3 – VCR Panel ............................. 238
Table 5-14: Problems and Attribute Comparison for User Test Task 4 - Remote ................... 240
Table 5-15: Problems and Attribute Comparison for User Test Task 5 - Barcode ................. 242
Table 5-16: Problems found in Pilot Study User Tests and GSE model Equivalents where predicted ........................................................................................................................................... 245
Table 6-1: Starting Templates for NGOMSL .......................................................................... 249
Table 6-2: NEC Potential Problems Summary ........................................................................ 304
Table 6-3: Samsung Potential Problems Summary .................................................................. 336
Table 7-1: Summary of Tasks in Main Study User Test ............................................................. 341
Table 7-2: Structure of Problem Tables used in this Study ....................................................... 343
Table 7-3: User Test GSE problems for Task 1 ....................................................................... 344
Table 7-4: User Test GSE problems for Task 2 ....................................................................... 345
Table 7-5: User Test GSE problems for Task 3 ....................................................................... 346
Table 7-6: User Test GSE problems for Task 4 ....................................................................... 348
Table 7-7: User Test GSE problems for Task 5 ....................................................................... 348
Table 7-8: User Test GSE problems for Task 6 ....................................................................... 349
Table 7-9: User Test GSE problems for Task 7 ....................................................................... 350
Table 7-10: User Test GSE problems for Task 8 ..................................................................... 351
Table 7-11: User Test GSE problems for Task 9 ..................................................................... 351
Acknowledgments

I would like to thank my supervisors Professor Geoff Drome and Cathie Sherwood for their tireless and unending inspiration, patience and support for the entirety of this odyssey. Specifically, I must acknowledge Geoff’s invaluable feedback and selfless dedication to this thesis and his GSE notation, under extraordinarily difficult circumstances. Words cannot convey the thanks owed to Cathie Sherwood for her dedicated guidance, thorough examination of my work, and the stern whip cracking I so frequently needed.

Thanks are also due to my mother and father. To mum, who literally chased obscure references to the edge of the universe and beyond, proofread until she could proofread no more, and provided so many other essential services she should have her own government department. To Dad, who though a competent accountant, never questioned the long-term investment that my lengthy study presented. And thanks also to Wade Robinson for devoting his free time to proofreading important chapters.

Thanks also to GSE boffins, Cameron and Dan who yielded so much of their time, without reward, assisting me with modelling matters, to usability experts Simon and Dan for their thorough analysis of the user tests and prompt results and to the dedicated transcribers Bernard, Llyne, Oisin and Holley who undertook the most tedious tasks of all. Much appreciation must be passed on to all thirty or so participants in the user tests who so willingly gave up their time.

In the end I dedicate this thesis to my supervisors and parents, who travelled along side me the entirety of this long and difficult road, stuck with me through thick and thin, and kept me going when times were hard. No amount of thanks can ever do them justice.
CHAPTER 1
INTRODUCTION TO THE STUDY

“In all approaches, design for usability requires considerable expertise and commitment to usability, neither of which is conventionally nor realistically available in the crucial early stages of technical design.”
(Thimbleby, Cairns and Jones, 2001:99)

1.1 The usability problem

Usability, or the quality of use (Bevan 1995b), is an important consideration when designing software and other interactive applications. Usability is typically measured by user testing. This involves examiners observing users physically interacting with a product, analysing the results and identifying usability problems to be fixed. User testing requires an application to be in a deployed and usable state for subjects to use (NITRD 2003: 19). This means that this testing can only be conducted well after the early stages of design or redesign. This is unfortunate, because the early stages of design present the designers with the freedom to make the most significant changes to the design of an application (Card Moran and Newell (1983) with the least expense (Karat 1997). It is little surprise then that the National Coordination Office for Information Technology Research and Development (NITRD 2003) for the United States government recently proposed the development of a predictive pre-deployment approach to usability as one of its current research needs.

This thesis proposes a predictive approach to usability design. This proposal is in accordance with the NITRD (2003) requirements for Human Computer Interaction and Information Management research needs. Thus the approach proposed in this study is based on the three NITRD requirements for a predictive technique. These are as follows:
1. A model of cognitive processes (human cognition and behaviour)
2. A total system model
3. Predictive properties which contribute to usability

This approach is not designed to replace user testing, which is widely recognised as an effective and valuable approach for evaluating usability (e.g. Ortega, Perez and Rojas 2003). Instead, the predictive approach proposed in this thesis, is intended to
predict as many usability problems at an early stage of design or redesign where major changes may be more easily made. It may also be used in situations where user testing is not a feasible option, due to time, experience or expense.

As a means of justifying this approach, the technique is tested in two different studies. These independent usability tests have been conducted in a pilot study on Video Cassette Recorder (VCR) operation, and a main study on mobile phone usability. The remainder of this chapter introduces the importance of usability, and designing for usability. It also introduces some of the approaches and techniques associated with usability design and why a successful usability related pre-deployment technique might prove valuable. The final part of this chapter outlines the structure of the thesis.

1.2 Pre-deployment and post deployment usability

Several major computer companies, including industry leaders like Microsoft and IBM continue to discuss how to make technology more usable (Ease of Use Roundtable, 1999). Their common goal is to improve the quality of the user's experience when using computer technology. The quality of the user's experience (UE), or 'quality of use', is better known as usability. Usability is a combination of the user's satisfaction, effectiveness and efficiency when using a product (NIST 2000). While the benefits of usability are well documented (e.g. Karat 1997, Myers, Hollan and Cruz 1996, Nielsen and Landauer 1993), improving the usability of a product is often complicated.

The alternative to solving usability problems is to prevent them. There are stages, prior to prototyping, where user-experience professionals can make a difference to the eventual user experience. These professionals include technical writers (who write the manuals), graphical designers (artwork and layout) and behavioural designers. Of most interest to this study are the behavioural designers, since they analyse, model and predict the user's behaviour, before user testing. Behavioural designers are usability professionals “who can create a cohesive conceptual model for the product, a model that is consistent, easy to learn and understand, and will form the basis for engineering design.” (Norman 1998:190). Behavioural designers are there to
anticipate the user's behaviour and provide a behavioural model that guides the system engineers to build a more usable product.

Unfortunately, a given project will not always have access to usability experts, especially where time, cost, and resources are limited (Nielsen 1993). In other situations, the system engineers may see little value in usability engineering and make little use of the experts available (e.g., Catarci, Matarazzo and Raiss 2002). Iterative and circular processes (as in Figure 1-1) aimed at improving the usability of a product may be expensive to apply. This is especially so in the early and crucial stages of design (Thimbleby, Cairns and Jones, 2001:99). Thus, experts in usability and its related field, (Human Computer Interaction (HCI), continue to look for ways to make their techniques, including behavioural design, more effective and accessible.

Since behavioural designers are not always available, this study proposes a process that integrates user behaviour and system behaviour into a common process. This process is designed to identify a number of usability problems early, by formalising usability principles that exist before user testing, into a tangible modelling process. This process makes use of a method based on cognitive engineering principles and another focused on software engineering. However, this approach is not limited to traditional software development and may be applied to other interactive products and technologies. The remainder of this chapter examines other approaches to usability analysis and design. It briefly argues the case for this approach and outlines the structure of this thesis.
1.3 Approaches to usability analysis and design

A product's usability and the experience of its user are largely dependent on its user interface (UI), those parts of a product that facilitate interaction between the product and its user. It is the user interface that has the most significant impact on the users' impressions of a product (Wiklund, 1994), and on the productivity and satisfaction of the user (PITAC, 1999). User interface design is an important part of HCI (Human Computer Interaction), a discipline focused on usability, the needs of users and enhancing the interaction between users and systems. A usable product is dependent on the behaviour of the interface and the usability of the interface is dependent upon whether it supports the user's needs and requirements. In the past, the quality of user interface design suffered because the people that built them lacked the knowledge and expertise needed to make their interfaces 'usable' (see Myers 1996).

Usability has been given several different definitions over the years, but is now generally recognised as a combination of effectiveness and satisfaction (Bevan, 2000), and a number of other qualitative terms such as 'attractiveness' (Ortega, Perez and Rojas, 2000). These definitions promote user testing as the most important usability technique, since satisfaction and similar principles are subjective in nature. (Olsen [Jr]. & Halverson, 1988) These subjective attributes are only reliably determined by having access to observe real users actually using the product. However usability testing on its own has been criticised for occurring so late in the process. Böckner (2000) for instance, argues that usability should not just be a stamp of approval, but should also apply to activities in the design process. Other prominent usability experts such as Norman 1998, and Nielsen 1993 acknowledge that some aspects of usability can be addressed before user testing. This is not to deny the importance or effectiveness of user testing. Instead, it clarifies that the entire design and development process offers opportunities to make the end product more usable.

Because usability is such an extensive concept there is no ‘single’ pre-test method to satisfy every aspect of the design and development process. There are too many contextual, cognitive and technical features, for a single method to address them all. In some instances, there are a significant number of alternatives available to perform a single task. For instance, task analysis, a means of extracting the user's tasks from a
set of requirements, has at least 41 different approaches (see Kirwan and Ainsworth, 1992).

There are a number of other factors that complicate usability design. Designers often have to make tradeoffs between different quality related principles such as Simplicity and Efficiency, which are sometimes in opposition to one another (e.g. Galitz 1997). An additional complication is that whenever a usability problem is fixed, it can also lead to new usability problems (Nielsen and Landauer, 1993). Despite this complexity, the design process offers a number of opportunities to influence the usability of the end product. To distinguish the differences between them, these techniques and processes are categorised as follows:

- Cognitive Engineering
- Requirements Elicitation, Requirements Analysis and Task Analysis
- Guidelines, Style guides and Heuristics
- Expert Reviews

**Cognitive engineering**

Cognitive engineering is an attempt to analyse, understand and predict user behaviour. Some analysts suggest that in the future, the need for actual users could be avoided completely. These analysts propose building complete models and simulations of human reasoning. At present, however, cognitive engineering is limited to less ambitious models of human behaviour such as GOMS. GOMS is a means of breaking down the user’s tasks into an organised set of Goals, Operators (e.g. key presses), Methods for completing tasks and rules for selecting between methods. This is a technique based on HCI and psychological principles, and is used for evaluating designs and predicting user behaviour. This technique, and its variations are described in detail in the third chapter.

Cognitive engineering predicts user behaviour through the construction of cognitive models of human behaviour and interaction. User behaviour is treated as ‘the acquisition and use of a cognitive skill’ (Bovair et al 1990: 7). This means that by producing a cognitive model, of what the user will need to know and do to complete a task, their behaviour may be predicted. Cognitive engineering does address the
different ways a user can perform a task and how long it takes to learn. Unique contextual issues and issues related to accessibility are not analysed by cognitive engineering. These are typically handled as part of the requirements elicitation and analysis phase, before cognitive engineering. Cognitive engineering techniques are often associated with task analysis, but there are many other task analysis techniques, some of which offer little guidance to ensure end-product usability.

Requirements elicitation, requirements analysis and task analysis

While a number of techniques are able to determine the system requirements, many do not adequately take into account the needs of the user (Carroll 2002). There are two important types of requirements (Richardson et al, 1998). There are the user’s task requirements, ensuring they can get the job done in an efficient and satisfactory manner. The other requirements are those that concern the logical structure of the system. These functional and technical requirements ensure the system can be modelled and implemented to support its purpose. Unfortunately, software is often criticised for ignoring the user’s needs and requirements in favour of its capabilities (e.g. Shneiderman 1997, Cooper 1995).

Actually obtaining or ‘eliciting’ the requirements is the first important stage of product design. Norman (1998) suggests that where field studies are used to determine the user requirements, anthropologists and sociologists should conduct them as part of a human-centred development team. The elicitation of requirements occurs before the behavioural design techniques proposed in this study. Since eighty percent of re-engineering software is involved in addressing unforeseen user requirements (Karat 1997), this first stage of design is extremely important. Several ways to obtain and analyse requirements are discussed in Chapter Three.

Guidelines and heuristics

Since the expertise of usability experts is not always available, there are a number of techniques intended to pass on this knowledge to the designer. Guidelines and style guides are sets of rules and recommendations designers may use when designing a system. These provide rules for the size of visible components, the colours used, how to prevent errors and other important issues. Guidelines are often criticised for being too vague (Henninger 2000), difficult to apply (Lewis and Rieman, 1994, Carter,
1999), easily outdated, and in some instances, too specific. While usability experts acknowledge the value of guidelines, some have introduced more general principles related to usability. One of the most prominent includes Nielsen's use of heuristics. These are a short set of key HCI related principles, which may be applied in the process of designing a product or evaluating a design. Despite criticism of heuristics as a poor alternative to user testing, it is defended as being "better than nothing" by its creator (Nielsen 1993), in situations where user testing is not feasible.

**Expert reviews**

Even when an HCI expert is present to provide a review of the design (e.g. the expert review method), there are still a number of problems. Methods that rely on judgement calls, such as expert reviews and cognitive walkthroughs, are often criticised for relying too much on a person's knowledge and opinions (e.g. Jacobsen, Hertzum and John, 1998). Also many of the above methods require usability professionals who are not always affordable or available.

Since expertise, time, expense and complexity all limit the application of usability techniques, a number of solutions have been suggested for encouraging designers to address usability. The next section summarises these approaches, and how the process documented in this study offers a new opportunity at making formal usability design more attractive.

**1.4 Making Usability and HCI more attractive**

There are five approaches to making usability design both more practical and attractive in design situations where it might otherwise be ignored. These are as follows:

1. Making a technique easier or less time consuming to use
2. Training designers in HCI and usability.
3. Automating the design and development process.
4. Extending an existing software engineering technique
5. Integrating software engineering and cognitive engineering.
Making a given technique easier or less time consuming to use

If existing techniques are proving difficult or time consuming, some analysts have proposed variations to existing techniques. These are aimed at making these techniques easier and more accessible. Nielsen’s heuristics, for instance, were proposed as an alternative to lengthy guidelines and extensive user testing, by condensing a large number of usability problems, into a small set of common principles. Nielsen’s (1993) heuristics are excellent rules of thumb, but specify no formal process for a designer to follow.

GOMS, a prominent technique for predicting and modelling aspects of user behaviour, has been adapted several times to make it easier. NGOMSL, or the Natural GOMS Language (Kieras 1997) and its computational form GOMSL (Kieras 2004a) are an attempt to make GOMS’ goal orientated technique more usable. Another GOMS technique, QGOMS (or Quick and Dirty GOMS), is no longer widely used (Beard et al 1997). NGOMSL is a little more flexible and is still used and taught (e.g. Gray and Mayfield 2002) However like other GOMS models, it is not so easily integrated with software engineering techniques such as UML and GSE.

Train the designer in HCI principles

There are suggestions that non-skilled developers should be encouraged to have an appreciation of usability techniques, especially those tasks that require “limited skills and resources” (i.e. Perlman 1996: 348). This is undermined by situations where systems engineers are reluctant to work with usability experts (Catarci et al: 2002). Training designers in usability principles can be time consuming and expensive and, like user testing, may or may not be a viable option.

Automating cognitive engineering and modelling techniques

A natural solution to a difficult design technique is to automate the technique by building a tool to support it. But this is no guarantee that the process will prove effective. Andre et al (2001) sum up this important problem by arguing that:

“the consistent use of a tool does not guarantee that the output of a usability evaluation will produce quality problem reports that communicate problems and causes precisely and suggest solutions for down-stream redesign activities.” (Andre et al, 2001: 132).
More recent GOMS tools, like Apex (John 2002) and GLEAN3 (Kieras 1999) demonstrate how the building of GOMS based models may be better facilitated by software. However, neither of these tools shares the integrated behaviour tree architecture of the GSE method proposed in this study.

**Extending an existing software engineering technique**

The opposite approach to changing a usability design technique, to encourage its use by software engineers, is to modify a software engineering technique to address usability. Recent initiatives to integrate the user’s goals with approaches to software engineering have concentrated on the Unified Model Language (UML). UML comprises notation for modelling different aspects of software design, employing nine standard diagram types including object-orientated class diagrams and use cases (Demurijan Sr and Pia, 2004). In order to accommodate the goals of Human Computer Interaction, there have been several recent proposals to integrate UML with ‘usability’ approaches. This is because UML models do not provide support for representing user behaviour (Dromey, 2002a). Adapting UML to overcome this requires additional or modified diagram types, since it cannot be done with UML alone (e.g. Arnowitz, et al: 2000, Scogings & Phillips: 2001, Baumeister, Koch, Kosiuczenko, and Wirsing, 2003). It is clear that UML is well suited for object orientated software design. However if UML is difficult to adapt to usability design, perhaps another software engineering representation might be more easily adapted.

**Integrating software engineering and aspects of cognitive engineering**

The final approach to make usability more practical for software engineers is to integrate compatible software engineering and usability related approaches. This is equally important to usability experts since user interface design is likely to be unproductive if the underlying system is not understood (Alm 2003). Cognitive engineering is primarily focused on the user’s behaviour. This is why some have suggested cognitive models should also represent the system (Green, Davies and Gilmore, 1996). This study proposes to do just that, by integrating software engineering and a cognitive engineering into a cohesive and effective modelling process. The next section introduces this approach.
1.5 A Predictive Approach to Usability

Two decades ago, Kieras and Polson (1985) attempted to make GOMS part of a wider system representation technique. Their efforts were limited by the GOMS models, which were available at the time. This study proposes integrating a software engineering technique called Genetic Software Engineering with the Natural Language GOMS approach.

1.5.1 Genetic Software Engineering (GSE)

GSE (Genetic Software Engineering) is a process proposed by (Drome 2002a, 2004c) for building software from its set of requirements, using an underlying notation based on behaviour trees. This technique is relatively recent in its development. Glass (2004) recently drew attention to the GSE technique by asking:

"There you have it. A fairly simple idea, accompanied by a kind of satisfying metaphor (software requirements integration, as the outcome of a jigsaw-puzzle-piece-process). I will ask again is this a revolutionary idea?" (Glass, 2004: 25).

GSE uses hierarchical behavioural trees that may be used at any level of detail to represent any number of components. The model is based on the assertion that a complete model of a system’s behaviour can be represented by its components, the changes in their states and the exchange of data between them. The model accommodates user behaviour by modelling the intended human participants as components of the system. These may be characterised as a single generic user, by job type (e.g. accountant) or as distinct individuals, depending on the detail of the requirements and needs of the analyst. The GSE technique offers a formal process for transforming system requirements into an effective model of the entire system, which is used as the basis for engineering the end product.

1.5.2 The Natural GOMS Language (NGOMSL)

NGOMSL or The Natural GOMS Language (Kieras 1997) is a variation of the cognitive engineering technique (GOMS or Goals, Objects, Methods and Selection). Kieras (1999) has since refined NGOMSL for use with his GLEAN3 software, and built a more recent computational form called GOMSL (see Kieras 2004a). In NGOMSL, the user’s behaviour is predicted by analysing the user’s goals. The goals
are broken down into a set of selection rules, methods for meeting the goals, and sets of operators (such as pressing a key) to represent the user’s behaviour. NGOMSL uses a more natural language as the basis for its model. It also includes several mental operators, in order to describe and analyse the user’s cognitive load. NGOMSL is hierarchical in nature, and like GSE, can be expanded to different levels of detail.

1.5.3 Comparing NGOMSL and GSE

Keiras and Polson (1985) set six requirements for a common representation of user and system behaviour. In order to justify the approach in this study, these requirements are discussed in relation to the common features of GOMS and GSE models.

Requirement 1: The representation should have well defined formal properties so that its correctness may be explicitly determined.

GSE and NGOMSL both have a formal process to follow when building them. The GSE process involves translating requirements, into behaviour tree representations (see Dromey 2004a) and then combining these representations into a single model. GSE has rules, notation and well defined modelling concepts for representing and integrating components and their states. The models can be checked for correctness against these rules and against other GSE models.

The NGOMSL method has four key components, the notation for which is described comprehensively in Kieras’s (1997) NGOMSL text and additional lecture notes provided by Kieras (2002). These are the Goals, Operators, Methods, and Selection Rules, each of which is formally explained in the Kieras’s NGOMSL guide. Kieras makes several recommendations on selecting high-level goals and outlines a formal process for refining NGOMSL models.

Requirement 2: The representation should make interactive systems easy to represent.

According to Dromey (2006: 96) the way we represent things often has significant consequences for "the complexity and relative difficulty of a task, the ease of understanding and changing what is represented and the likelihood of making
mistakes.” It is important therefore to pick a representation that while functional, is not excessively difficult to apply.

The GSE notation is extensive but simple and employs only one type of model (the behaviour tree) to represent the behaviour of an interactive system, though there are clearly other design activities that will require other techniques. In GSE, the users’ behaviour is represented in the same way that system components are represented because the relevant human participants are also represented as components. The relative simplicity of the GSE notation makes GSE models relatively easy to build. GSE permits events, decisions, state realisation, user input and system output to screens and other devices. The NGOMSL model describes the user’s behaviour in a natural language format. This makes it easier to read than other GOMS formats. NGOMSL cannot represent system behaviour as extensively as GSE, since it deals primarily with user-system interaction and does not address interaction between components of the system, or express adequate formal constraints for changes in system state where the user is not involved. This is why this study proposes using it as the basis for GSE models.

Requirement 3: The representation should be modular, so that as much or as little of the device can be represented as desired.

Both NGOMSL and GSE allow the designer to model their system to a chosen level of detail. In GSE, designers may collapse a fragment of behaviour to a single node or expand nodes to include much more detail if necessary. NGOMSL supports ‘dummy stubs’, place markers for complex behaviour that the analyst decides not to break down into any further detail (see Kieras 1997). Conversely, NGOMSL may also be described to the level of a set of atomic operations such as hand movements if that kind of detail is required.

Requirement 4: Hierarchical control or structural relations should be easily represented.

NGOMSL and GSE are both hierarchical in nature. In GSE, an interactive system may be partitioned into a series of nodes, which in turn may be partitioned into more detailed behaviour trees. A series of different notations are used to represent relations between nodes. The most common are component states and changes in state.
represented by square brackets and question marks respectively. Rules for the common conditionals “And”, “Or”, “Not” and other conditionals are provided. Behaviour trees also support the logic and formal constraints of programming languages, and can represent iteration, sub methods and other operations in the trees themselves. NGOMSL has simple selection rules for relating methods and goals to one another. It uses a smaller set of operations to describe the users’ behaviour than GSE. The NGOMSL rules for choosing between alternatives are easily represented in GSE.

**Requirement 5: The representation should not be committed to any particular hardware or software implementation.**

The NGOMSL and GSE processes are completely independent of software or hardware. GSE models have been used to describe the operation of satellites, microwave ovens and the progress of murder cases. NGOMSL has been used to describe text editors, the operation of video recorders and the operation of combat aircraft.

**Requirement 6: It should be easy to represent features of the system that have psychological implications.**

GSE is a flexible notation method, whereas NGOMSL is based on psychological principles outlined by Card et al (1983), the creators of the original GOMS model. NGOMSL makes cognitive operations explicit, and is able to represent access to long term and working memory. While this study does not model complex psychological processes in detail in GSE, it does address the key constructs of NGOMSL models (goals, operators methods and selection rules) and design attributes, which have psychological implications.

**1.5.4 Justifying an NGOMSL/GSE approach**

Ultimately the method proposed in this study will be judged on how it advances the cause of usability design. Dromey (2002a: 36) sets three conditions for a new method that need to be met:
1. To provide an advance over existing methods rather than just come up with other alternatives.

The NGOMSL-GSE method is intended to advance existing methods, which do not provide a complete representation of an interactive system. This approach is designed to consolidate the goal-orientated aspects of cognitive modelling with a software engineering model. The single underlying behaviour tree notation in GSE is intended to be less cumbersome than employing a swathe of multiple diagram types such as those offered by UML notation.

2. To build on and exploit as far as possible the best-proven ideas about design that are currently available.

It is well documented that usability engineering offers significant and tangible benefits when compared against the costs. It is also well understood that usability, cognitive modelling and usability principles, when addressed effectively, raise the quality of the end-design. While it is implausible that every possible design consideration can be addressed by a single approach, consolidating system design with well recognised design principles should better address usability issues neglected by other system orientated processes. How this compares against other approaches, focused on other or more specialised aspects of system development, is a question for further consideration.

3. To identify weaknesses in existing design methods and seek to overcome them to identify the scope and any limitations of the new proposals for design.

The NGOMSL-GSE approach is based on the following assertions:

- Some aspects of usability can be addressed without user testing.
- Software engineering models alone, such as those based on UML, do not always address usability issues adequately.
- User focused models based on cognitive engineering principles are unlikely to provide representations that adequately represent the system, or are suitably compatible with software engineering techniques and representations.

The next section outlines the contributions of this thesis.
1.6 Contributions of the thesis

This thesis sets forth the hypothesis that a pre-deployment usability design technique based on the GSE and GOMS analysis techniques and a set of HCI properties can formally identify a number of usability problems without user testing. If the model is successful, the new technique will be able to identify usability problems, many of which are typically only revealed by testing with actual users.

1.6.1 Aim of the research

The aim of this thesis is to build a total system model based on both cognitive modelling principles and system modelling that allows the designer to predict usability problems prior to development. This technique is not designed to replace user testing, but to identify as many usability problems before deployment where major changes can be more easily made with less expense. This aim will be achieved by providing the following:

- The establishment of a set of quality attributes, modelled in GSE, in order to better identify potential usability problems. These will be based on established design principles, and applicable to different tasks and devices.

- The proposal of a new integrated design technique based on cognitive and task modelling principles, and software engineering representations, which can apply the aforementioned attributes to predict usability problems prior to user testing. This approach may be used to design software, hand-held devices, phones, web pages, and any other device or application that has a user-interface.

- The definition of a transformation technique for using a NGOMSL task analysis, and its goal based structure as a way of building GSE models that represent both the behaviour of the system and the user. Additionally, aspects of NGOMSL will also be used in GSE models of existing systems to better represent the user’s behaviour alongside the behaviour of the system.

In order to prove the merit of the new approach, it will be tested against the previously stated criteria in 1.5.3 and 1.5.4. Additionally, Bastien et al (1999)
suggests that ‘reliable’ methods need to be consistent between evaluators, and provide coherent results. To do this, this thesis will be tested against several user tests employing multiple examiners.

1.7 Description of Chapters

In order to validate approach proposed in this study, this thesis is structured across the following seven chapters:

**Chapter Two - Human computer interaction and usability**

The second chapter discusses important concepts related to human computer interaction and usability. The purpose of this chapter is to identify the fundamental principles that contribute to a product’s usability. It begins by examining the relationship between user interface design, usability, quality and the users themselves. It also critiques the major usability techniques in detail. This chapter concludes by categorising and identifying quality attributes which directly influence usability.

**Chapter Three: Modelling and analysing behaviour**

The third chapter examines two methods that form the basis of the NGOMSL-GSE approach. This chapter begins by discussing the role of task analysis, design and modelling in relation to design. It discusses cognitive modelling and GOMS approaches in detail, to demonstrate how cognitive modelling and changes in GOMS have led to more practical approaches like NGOMSL. This chapter also introduces the NGOMSL model building process. The latter parts of the chapter, discuss how software engineering and software design processes address usability issues. This chapter concludes by comparing GSE behaviour trees against the Unified Modelling Language, and which of these two notations is more compatible with a NGOMSL representation.

**Chapter Four: Methodology**

This chapter outlines the procedures used for both studies. This includes a concise description of how to build GSE models and how this approach may be applied to find usability problems. The third section contains GSE representations of usability problems based on the attributes discussed in Chapter two, which may be used to find usability problems. The latter sections of the chapter document the procedures for the
user tests carried out in the pilot and main studies. This includes the choice and selection of subjects, devices and examiners, a map of the test environment and other test features as specified in the Common Industry Format for user testing (see UIST 2001).

Chapter Five: VCR – Pilot study

This chapter is the first demonstration of the NGOMSL-GSE method. Since the timer recording capability of Video Cassette Recording devices (VCRs) is often criticised for its lack of usability in relation to timer recording, the pilot study was conducted on the timer recording interfaces for two VCR devices. This chapter discusses an existing NGOMSL model of VCR behaviour, applies a GSE conversion, builds an As-Is model of the VCR’s behaviour in GSE based on the goal-orientated structure of the NGOMSL model and then compares the usability of several GSE models against the results of a VCR user test. The last section summarises and discusses these results and how successful the NGOMSL-GSE approach is at identifying usability problems.

Chapter Six: Main study – Mobile phones

The main study is similar to the pilot study, but concerns emerging mobile phone technology. A major difference between this chapter and the pilot study is that an original NGOMSL model of mobile phone behaviour is built using the NGOMSL method. As-Is models of the two phones’ behaviour are also built, based on requirements specified in the NGOMSL model. In the second half of the chapter, GSE representations of several mobile phone tasks are analysed for potential usability problems using the GSE attribute representations documented in Chapter Four.

Chapter Seven: Results and discussion

Chapter Seven analyses the results from the GSE analysis in the main study against the findings of the user tests. Each problem found in the user test is compared against the problems predicted in Chapter Six according to the level of severity and the usability related attributes involved. This chapter concludes with a summary and analysis of the data as it relates to the identification of usability problems.
Chapter Eight: Conclusion

The final chapter summarises and clarifies the findings of this study. It also suggests possible improvements to the NGOMSL-GSE technique and raises several other issues and recommendations which may be addressed by future studies.

1.8 Summary

This chapter has provided reasons for integrating a software engineering model with a user-orientated cognitive model by consolidating NGOMSL and GSE with the aim of better identifying usability problems. Chapter Two discusses usability, its importance to design, and the concepts, principles and techniques relevant to its practical application.
CHAPTER 2
REVIEW OF THE LITERATURE:
USABILITY AND ITS PROPERTIES

The lack of consistent, specific terminology in the world of software design frustrates interface designers enormously. Without precise terminology, we are forced to speak in vague generalities and hand waving. Without clearly differentiated terms, we accidentally group things in the wrong places, overlook significant facts, and inadvertently mistake the bad for the good (Cooper, 1995:5).

2.1 Usability design and quality

The main aim of this chapter is to identify a set of properties, which may aid in predicting usability, prior to the deployment of an interactive application. There are three main parts to this chapter. The first part involves a detailed discussion on the background to usability and its relationship to key concepts such as user interface design, product quality and HCI. The second part discusses how usability is currently measured and defined. The final part discusses properties associated with usability and how they might be used to predict usability without user testing.

2.1.1 The value of HCI and usability

The term ‘usability’, though over a decade old, is far younger than HCI, the study of interaction between humans and machines. HCI is a significant field of study. According to De Graaff’s (2003) index there are no less than 30 conferences concerning HCI scheduled for 2004. It also notes that there are over 210 groups working on HCI for research and commercial purposes across 22 different countries, with 11 based in Australia. Though there is clearly significant interest in HCI, there are a number of reasons for addressing HCI and usability issues.

Productivity

The US government’s submission on Information Technology for the Twenty First Century (PITAC: 1999) asserts that over 12% of an American computer user’s time is wasted because they cannot tell what a computer is doing. This demonstrates that an incomprehensible system wastes a significant amount of the users’ time and impedes their productivity.
Standards compliance

According to Rada (1996) standards are the specifications recognised within an organisation. There are, however, organisations, which produce standards for the entire community. The European Community, has tackled the compliance issue by introducing mandatory user interface standards for software (see Bevan 2001). Other similar standards are necessary to obtain ISO (International Standards Organisation) certification (Refer to Schrier, Williams, MacDonell, Petersen, Strijland, Wichansky, and Williams (1992:635). Thus, designing high quality user-interfaces is no longer voluntary in many instances.

Organizations such as the EEC and ISO are not the only sources of standards. A company may enforce in-house usability standards for their computing personnel (Rada 1996). While few deny that standards are a positive step, Buic (1999) warns that organisations may misuse standards by denying ‘certification’ to those who don’t adopt them.

In order to keep the terminology in this study consistent, the following definition by (Rada 1996:19) will be used to define a standard:

**Standard:** “a specification recognized within an organization”

The high cost of reengineering

According to Karat (1997) only 20% of software maintenance is spent on the identification and elimination of bugs. The other 80% is spent addressing unforeseen user requirements. Unfortunately, re-engineering software can be both expensive (Nielsen and Landauer, 1993, Myers, Hollan and Cruz, 1996, Karat, 1997, Nielsen and Phillips, 1993) and time consuming (Sawyer, Flanders and Wixon, 1996, Bekker and Vermeeren 1996). In order to avoid these costs, usability and user interface problems need to be tackled early in the design process.

Costs versus benefits

According to Myers, Hollan and Cruz (1996) applying HCI techniques and processes can dramatically reduce costs. Nielsen and Landauer (1993) for instance, argue that the cost-to-benefit ratio can be as high as 1 to 247. In practical terms this means an
investment of $US33000 to obtain nearly $US8,300,000 worth of benefits for large-scale projects. For small projects the ratio is less compelling, but still valuable, with a cost benefit ratio of approximately 1 to 4. In other instances, after performing a comprehensive analysis of a completed project, Karat (1997) found for each $US1 spent, there was a $US10 return. Thus, there is clearly much to be gained for each dollar spent addressing HCI issues.

2.2 Key usability concepts

Usability is an important facet of HCI, but there are a number of related concepts, which are also important. Many of these concepts are critical to understanding human computer behaviour. These important concepts include the design of the user interface, the theory of mental models and how people differ in their behaviour when using machines. This section provides a brief discussion on these topics.

2.2.1 The user interface

According to Wiklund (1994) The user interface is:

"an excellent place to focus quality improvement initiatives, because of its high visibility and the fact that so many users base their first and lasting impressions of a product on its interface. " (Wiklund, 1994: 9)

The user interface is the medium by which users interact with the mechanisms, and systems they use. The interface has been defined the ‘contact surface’ of a thing by Laurel (1991). This is a very broad definition, and encompasses everything from door handles to the buttons on video recorders. However the term is typically applied to interaction with computer technology. According to Mandell (1997:14) the user interface not only encompasses the presentation of information and facilitation of interaction with the user, across both the hardware and software, but also includes the “whole system experience” between user and computer. This “whole system experience” is better known as the “user experience” and its relationship to any definition of the user interface is described as follows:

"The user interface can arguably include the total "user experience," which may include the aesthetic appearance of the device, response time, and the content that is presented to the user within the context of the user interface." (Whatis.Com 2003).
It must be noted that the authors like Armitage (2003) and McClelland (2005) make
distinctions between user interface design, and what they call 'experience design.'
However, for the purpose of this study, this study adopts the more inclusive view
expressed by Marcus (2002) that such a distinction is largely unnecessary.

In Mandel's (1997) definition for the user interface it is evident that it includes
peripherals (e.g. the mouse, monitor and keyboards), and any other device with which
the user interacts. Another significant feature of Mandel's definition is that it also
includes any manuals or documentation used as part of the interaction. This is
consistent with a concept that Norman calls "the system image", which includes any
and all parts of both the system and relevant content with which the user interacts
While this study examines properties that contribute to usability, it does not analyse
the structure of external learning and reference material such as manuals. Thus, for
the purposes of this study, the following definition is used:

**User interface (UI):** "A computer-mediated means to facilitate
communication between human beings or between a human being and an
artifact. The user interface embodies both physical and communicative
aspects of input and output, or interactive activity. The user interface
includes both physical objects and computer systems (hardware and
software, which includes applications, operating systems, and networks."
(Marcus 2002: 19)

### 2.2.2 Recent interface developments

The growth of the Internet and the World Wide Web (WWW) has radically expanded
the ways in which users interact with their computers. Currently content is presented
in Internet browsers like Microsoft Internet Explorer and Netscape Navigator via a
number of different page formats that are supported by HTML, the Hypertext Mark-up
Language. The presentation of interactive content, via web pages, has raised the
question: are web-pages user-interfaces in their own right? Web designers are not in a
position to reengineer the browser in which their pages appear. But as web pages
become more and more powerful, through embedded code, extensible mark-up
languages (XML) and new presentation technologies, (e.g. Scalable Vector Graphics,
and Flash animations), interaction is becoming both more powerful and more
complicated.
It is increasingly clear that web pages are user interfaces in their own right. For instance, Shneiderman, Nielsen, Butler, Levi and Conrad (1998:92) argue that a website is a user interface because: "its design must consider many of the traditional usability issues in interaction design." But the rapid growth of the internet and the proliferation of web-page design tools, have given more users the option to make their own pages.

Another development of e-commerce is the proliferation of web sites that support online business transactions such as on-line banking and auction sites. Despite the fact that these sites provide a means of revenue, e-commerce is also subject to poor UI design. According to Spool et al. (1998), who studied 15 different commercial web sites, users could only find the information they were looking for 42% of the time. Another study by Zona Research (1998) on e-commerce sites, called the “Shop Until You Drop Study” estimates one person in three finds it extremely difficult to locate a specific product. Most of these problems, were caused by poor interface design. In fact, an e-trust study by Cheskin Research and Archetype Sapient (1999) found that good user interface design was a major factor in determining whether users were able to trust their web sites.

A more recent development in user interface design is the proliferation and development of mobile phone technology and personal organisers. These rapidly evolving devices, provide access to functions traditionally associated with computer systems. Myers, Hudson and Pausch (2000) call this the age of ‘ubiquitous’ computing. They argue that these new small handheld technologies will cause major changes to both the user interface and the tools and methods required to design, build and evaluate them. The usability of mobile phone devices with sophisticated functionality is examined later in Chapter Four.

2.3 Understanding and modelling user behaviour

While designers may design the behaviour of the system, it is not possible to design human behaviour. Several HCI analysts have spent considerable time analysing the way humans interact with machines, so that human behaviour may be both understood and anticipated. The NITRD (2003) research on requirements for HCI demonstrates
that there is still a significant need for a means of predicting user behaviour. This is why HCI has strong connections to psychological principles. Much of this analysis has involved the development of models related to human cognition and behaviour. Based on several definitions, models are defined in this study as follows.

**Model:** "a physical, mathematical, or otherwise logical representation of a system, entity, phenomenon, or process" ... "using some type of formalising consisting of variables and the constraints between these variables." "A model is also a description of the generic structure and meaning of a system," an abstraction capturing the essential aspects of a system (depending on the intent) and ignoring some of the details.


An early and significant means of modelling the behaviour of users was proposed by Card, Moran and Newell (1983) in their text, "The Psychology of Human-Computer Interaction". Their GOMS technique is a means of decomposing the anticipated behaviour of the user into a set of operations, methods and choices, based on two principles, which imply that humans behave in a goal-orientated manner.

### 2.3.1 Mental and user models

In 1988, Norman, a vocal critic of poor interface design, introduced his text, "The Psychology of Everyday Things." Norman’s text looked at how designers, engineers and users differ in their understanding of a system and how this contributes to design. To highlight the differences between these perspectives, Norman proposed models of human cognition and perception, the most important of which is the mental model.

Norman’s models are classified as follows:

- **The mental model:** The model created by the user in order to understand a system. It may be incomplete and will evolve as the user learns more about a system. It allows them to understand how the system works and determines how they interact with it.

- **The system image:** All the components of a system with which the user comes into contact, essentially all the parts of the user interface. It also describes how the computer looks and the content of any information provided to the user.

- **The conceptual model:** "a high-level description of how a system is organized and operates". It is an "idealized view of the how the system works" which is "the
model designers hope users will internalize”, “the ontological structure of the system” (“the objects, their relationships, and control structures”) and “the mechanism by which users accomplish the tasks the system is intended to support.” (Johnson and Henderson 2002: 26 – 27)

Mental model should not be confused with “user models” described by Roberts, Berry, Isensee and Mullally (1997) as “knowledge in the user’s head”. User models are not models conceived in the users mind, but are instead:

“an abstract representation of the user that models those aspects of user behaviour and knowledge needed by the interface.”

Vaubel and Gettys (1990:97)

The implication of Norman’s three types of model is that the conceptual model and system image should be as similar to the user’s mental model as possible in order to ensure good design. This is supported by Jacquot and Quesnot (1997: 50) who argue that a ‘good’ user interface can be defined as “one which offers images coherent with the mental model of users and a structure of commands coherent with the users’ problem-solving procedures. ”

Psychological models are not without their critics. Young, Green and Simon (1989), for instance, downplay the value of using models of the user. They argue that “psychology is not in a position to provide ready-made user models off the shelf, and that anyway such models are not necessarily suitable for the role intended for them” (Young, Green and Simon, 1989: 18). Young et al’s criticism is that psychological models are not necessarily a practical means for good design. Additional problems with mental models involving critical systems are also discussed in Besnard, Greathead and Baxter (2004).

Young et al’s argument also demonstrates the differences between some HCI experts on the value of psychological models as a basis for design. According to Green, Davies and Gilmore (1996), there are three different approaches to psychology and its impact on user interface design. The first of these, the system view, proposes that good design is reliant on the ‘common sense’ psychology of the designer. Young et al’s arguments are consistent with this view. The second view, the psychological view, proposes that cognitive psychology is a valuable and effective part of HCI. This
view is consistent with the use of psychological techniques, such as Card et al’s GOMS model. The final view, the interactive view proposes that human-computer behaviour is not merely a sequence of inputs and outputs, but is far more complicated.

The technique proposed in this study, seeks to anticipate some aspects of user behaviour and is consistent with the psychological view. However this study also recognises that human-computer behaviour is complicated and cannot be modelled in its entirety. The study also acknowledges that designers will also have to rely on common sense as part of the design process. So all the viewpoints above are considered valid to some degree.

2.3.2 Human diversity

Modelling a user’s anticipated behaviour is made more complicated by the differences between individual users. According to Nielsen (1993:43) the ‘user’s task’ and their ‘individual characteristics and differences’ are the two most important issues related to usability. In other words analysts need to understand both what a user has to do (their tasks) and who the user is, in terms of expertise, age and other factors, which influence their performance.

But should users be classified? Rubin (1994) argues that this is an essential part of conducting user testing, the most accurate means of identifying usability. So what about design? Martin and Eastman (1996) believe that designers can use their intuition where the tasks are well understood. Rubin also adopts the psychological

<table>
<thead>
<tr>
<th>Table 2-1 Characteristics of Experts and Novices</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Expert</strong></td>
</tr>
<tr>
<td>They possess an integrated conceptual model of a system.</td>
</tr>
<tr>
<td>They pay less attention to surface features of a system.</td>
</tr>
<tr>
<td>They organise information more meaningfully, orientate it toward their task. They possess knowledge that is ordered more abstractly and more procedurally.</td>
</tr>
<tr>
<td>They are better at making inferences and relating new knowledge to their objectives and goals.</td>
</tr>
</tbody>
</table>

Adapted from Galitz, 1997: 48 - 49.
perspective by recognising the value of user classification and common sense design decisions.

When classifying the user, analysts often make the distinction between ‘novices’ and experts. Galitz, for instance, provides a number of characteristics for distinguishing expert users from novice ones (Table 2-1). Experts are also able to organise and apply information more effectively (Zimmerman and Risemberg 1997 and Glaser 1992), and apply learning and self-improvement strategies (Erriscson and Charness 1994, Cleary and Zimmerman 2000). Novices however can be impeded by low-level details (Galitz 1997, Buxton 1986) that experts can handle more automatically. An excellent summary of expert capabilities can be found in Bransford, Brown and Cocking (2000).

Other factors which may be used to classify novices and experts include, computer literacy, system experience, experience with the application in question, task experience, experience with other systems, education, reading level, typing skills and native languages (Galitz 1997). Many of these are specified in the CIF guidelines (2001) for usability testing.

In practical terms, many design and evaluation techniques are criticised for ignoring both human diversity and context (i.e., see Buie 1999). How designers address human diversity varies, but for all their differences, Nielsen, Murray, Galitz and Martin and Eastman, all agree that human diversity is an important issue.

There are many other issues related to human diversity and difference, beyond the scope of this study. These include socio-economic factors, cultural and linguistic differences (Marcus 2003), gender (Cassell 2003), age (Czaja and Lee 2003, Bruckman and Bandlow 2003), and visual, physical or aural impairment (Sears and Young 2003, Jacko, Vitense and Scott 2003).
2.4 Usability and related definitions

This section discusses the practical aspects of usability: what is it, and how do you use it? According to Butler (1996:60) usability is measured empirically "by observing and recording actual user performance of frequent or critical tasks on the system."

Carter (1999) argues that if a system lacks usability, then the user will either not use it, or will not use it well. Usability is clearly a significant measure of the user interface. The large number of usability laboratories across the globe demonstrates the significance of usability (Bevan 1998). At its core, usability is a desirable property and therefore a type of quality according to Dromey's (1998) definition. But unlike the technical aspects of design, usability focuses almost entirely on the interaction between the user and the system.

Some time ago, 'usability' replaced the term 'user friendly' (see Butler 1996)

Usability is now used to classify a wide range of techniques focused on improving and facilitating the interaction between user and machine, through design and testing.

The practical application of usability is divided between two different views. One group believes the appropriate way to identify usability is to watch actual users. This group includes Bevan (1998), Ortega Perez and Rojas (2003) and Butler (1996). The other group believes that many aspects of usability can be anticipated or predicted by the designers before user testing. This group is best represented by Nielsen (1994) who proposed a set of heuristics for designers to use to evaluate the usability of their product. The first group can be described as post-implementation usability, since usability can only be determined with a working system. The pre-test group however argues that many aspects of usability can be determined before user testing.

These two groups have different definitions of usability. For instance, Ortega, Perez and Rojas (2003: 3) of the post-implementation group, define usability as "the capacity of a software product to be attractive, understood, learned and used by the user under specific conditions". In other words usability is only evident when the software is being used. Bevan defines usability as the 'quality of use'. He describes the quality of use as follows:
“The effectiveness, efficiency and satisfaction with which specified users can achieve specified goals in specified environments” (Bevan 1995:4).

However Bevan does concede that it is theoretically possible to evaluate the usability of a product by using attributes. He suggests that style guides, general usability principles and checklists may be used to both evaluate usability and address usability before user testing.

So what are the principles that contribute to good design? Some years ago, Nielsen (1993) created significant debate by introducing a set of heuristics that correspond to the usability of software (also discussed in Gray et al 1995). By examining 259 usability problems, Nielsen was able to identify ten generic heuristics for designing software to be ‘usable’. The heuristics demonstrated that some attributes were self-evident in design. Nielsen encouraged designers, who do not have the time or expertise to conduct tests with actual users, to use these heuristics. Thus, Nielsen’s heuristics provide a ‘better than nothing’ approach for identifying usability problems. More importantly they also demonstrate that some aspects of usability exist before user testing.

Based on the claims of Bevan (1998) and Nielsen (1993), this study adopts the following definition for usability:

Usability: the measure of how successfully the system facilitates the user in effectively, efficiently and satisfactorily meeting their goals, through interaction with the system.

Understanding usability is one aspect, using it for practical purposes is another. The next section describes the terminology for addressing usability as part of the development process.

2.4.1 Usability engineering, design and objectives

Usability Engineering covers almost all the techniques that relate to usability. According to Ivory and Hearst (2001: 473) usability engineering techniques include
methods for testing, inspection, inquiry, modelling and simulation. Thus the definition for usability engineering is as follows.

**Usability Engineering:** the process/discipline of using techniques aimed at testing, modelling or making a more ‘usable’ system at any stage of the development process.

In many instances usability problems correspond to an existing system. These systems may be ‘fixed’ or completely rebuilt. When the usability team must build the new system based on the structure of the old one, usability re-engineering is more appropriate. This is defined as follows:

**Usability Reengineering:** the process/discipline of using techniques aimed at improving the usability of an existing system.

While usability engineering is a very broad term, some techniques can only be used for the design (or redesign) of the system. Bearing this in mind, the following terms define the purposes and processes associated with designing for usability. These are defined as follows:

**Usability Design:** Designing a system to usability principles, guidelines, and or standards.

**Usability Objectives:** Any objective, design rule, principle or attribute that, when implemented, supports the usability of a product.

### 2.4.2 Usability testing, inspection and evaluation

User tests (or usability tests) are methods, which involve the actual observation of users (i.e. Ivory and Hearst 2001). Jacobsen, Hertzum and John (1998: 255) suggest that while there is no standardised procedure for usability tests, they are typically conducted in controlled environments and include a number of sessions involving a user working on set tasks while thinking out loud. There is a current standardised procedure for user testing, which may be found in the common interface format CIF for user testing. There is also an XML related format under development for reporting
user-testing results, presently available (Chisolm and Palmer 2002). For the purposes of this study the following definition is used for user testing:

**Usability Testing or User Testing:** a test involving the actual observation of users, after which the researcher analyses the observations looking for usability problems.

User testing requires a working system (or prototype) for the test subjects to use. However some usability techniques need neither users nor prototypes since they rely on some form of assessment criteria to assess the usability of a design. These are known as inspection methods, and are defined as follows:

**Usability Inspection:** the use of heuristics, guidelines or other criteria to find usability problems (Ivory and Hearst 2001, Nielsen 1993) without observing actual users.

There are other methods like GOMS, which attempt to anticipate and predict how a user will use a proposed design or system in order to determine usability measures. These techniques, which are of great importance to this study, are known as Model Based Evaluation and are defined as follows:

**Model Based Evaluation:** “using a model of how a human would use a proposed system to obtain predicted usability measures by calculation or simulation.” Kieras (2002: 1140).

User testing, usability inspection and model-based evaluation, all share a common purpose that is to evaluate the usability (or related factors) of a design or system. As such they can all be classed as usability evaluation techniques. Usability evaluation is defined as follows:

**Usability Evaluation:** the use of any method that provides a measurement of usability or seeks to identify usability problems, at any stage of the development process, including usability testing, inspection and model based evaluation.
2.5 Usability engineering techniques

Usability testing is described as the best way to identify the usability of a system. Unfortunately usability studies are expensive and time consuming, and it a mistake to rely on a high skill level that few possess or are prepared to pay for (Nielsen in Gray, 1995). In order to determine which attributes correspond to usability, it is important to begin looking at techniques that address usability issues before deployment and user testing. These techniques include guidelines and heuristics and cognitive walkthroughs. This section discusses these techniques and their role in usability engineering.

2.5.1 Guidelines and heuristics

Guidelines are rules and recommendations for designing a user interface. They are similar to heuristics though heuristics tend to be more general rules of thumb (Lewis and Rieman, 1994: 4:19). Examples of guidelines can be found in the following documents:

- Guidelines for Designing User Interface Software (1986)
- Ameritech Graphical User Interface Standards and Design Guidelines (1992)

Guidelines provide designers with design rules, specific to different aspects of development. There is some argument between what constitutes guidelines, and what constitutes standards. Buie (1999) argues most standards are just collections of recommendations, which may also be called rules, requirements or guidelines (Buie, 1999: 38).

In contrast, Schumacher (1992) argues that statements that are clearly ‘required’ are standards. Guidelines, on the other hand, are rules based on sound information and well-recognised design practices, but are considered to be less necessary. While these guidelines are mandatory standards for the organisation; they are clearly not mandatory for anyone outside the company.
Detweiler and Omanson (1996) clearly distinguish between standards and guidelines in their guidelines for website design. Table 2-2 contains their rules for the appearance of icons on web pages. The first two guidelines are standards that must be followed within their organisation. The last two guidelines can be considered recommendations.

There are two uses for guidelines. The first is as a reference when designing software. The second is as a checklist, where compliance to guidelines can be measured by checking which guidelines have been adhered to, and which have not. It is generally accepted that adherence to many of these guidelines can prevent some types of usability problems. This is of course dependant on the quality of the guidelines, and the skill of the development team in applying them.

Like any technique, the use of guidelines and heuristics has a number of advantages and disadvantages. The advantages of guidelines are that they,

1. Can be quick. (Jeffries, Miller, Wharton and Uyeda, 1991)
2. Do not require an HCI expert. (Buie, 1999 and Nielsen in Gray, 1995)
5. Can be used as an evaluation technique. (Morkes and Nielsen, 1998)
8. Have been proven to improve usability

Unfortunately, guidelines also have a number of disadvantages, which can make them undesirable. They are often seen to be:

1. Too big. For instance, Smith and Mosier’s (1986) guidelines consist of more than 300 pages
2. Hard to apply (Lowgren and Nordquist, 1992)
4. Incomplete. (Ratner, Grose and Forsythe, 1996)
5. Unable to keep up with technological change. (Bevan and Holdaway, 1993)
6. Too general and vague (Carlshamre, 1994: 202)
8. Deliberately ignored. (Borges, Morales and Rodriguez, 1996: 278),
9. Unable to identify many problems that concern the task sequence. (Lowgren and Nordquist, 1992)
10. Significantly different to other guideline documents. (Ratner, Grose and Forsythe, 1996)
12. Take a long time to learn (Jeffries, Miller, Wharton and Uyeda, 1991).
13. Able to miss serious problems (Jeffries et al 1991)
14. Of limited assistance to less experienced people. (Karat, Campbell and Fiegel, 1992; Tetzlaff and Schwartz, 1991: 329)

Thus, the use of guidelines has significant number of both positives and negatives. Perhaps the most significant limitation is that guidelines are not designed to find problems in the task sequence of an interface. Since performing tasks is such an important part of the interaction, guidelines are limited in their application. Guidelines and heuristics can be seen as valuable analysis tools but they will not address every factor that contributes to the usability of an application. However, there are other
techniques designed to focus on the tasks the users must perform. The best known of these is termed a "cognitive walkthrough" and is discussed in more detail in the next section.

2.5.2 Cognitive walkthrough

Cognitive walkthrough (CW) is a technique that focuses on learning and task completion. In a CW, an expert analyst "walks" through a design based on a set of pre-arranged "steps" (Virzi 1997). The sequence for this procedure is as follows:

1. Choose the specific task an interface must support
2. Determine one or more correct sequences of actions for that task.
3. The analyst examines these sequences in relation to the context, and whether a "hypothetical" user will be able to choose the appropriate action.

   (Lewis and Wharton, 1997: 717)

This method is used to determine whether a user will know what to do based on feedback and information from the interface (Blandford and Rugg, 2002). CW is a method to support design and development but not for validating the design (Lewis and Rieman, 1994).

According to Lewis and Wharton (1997), cognitive walkthroughs are different to "predictive" design techniques, because the walkthrough requires an existing sequence of actions. They also suggest that an additional limitation is that cognitive walkthroughs do not inform the designer about what happens when users deviate from this fixed sequence. While cognitive walkthroughs can be used on a prototype, this is not essential, since they can also be conducted on detailed design descriptions (Lewis and Rieman 1994).

Even though the CW method is relatively easy when compared with methods like GOMS; Lewis and Wharton argue that taking the time to learn the GOMS method is preferable to only using CW. Hudson et al (1999:100) tries to address these issues by integrating CW into a GOMS related technique called KLM (discussed next chapter).
CW, according to Lewis and Wharton (1997), is not worth conducting if it can't be conducted quickly, since the advantage of this method is the speed compared with slower but more comprehensive methods. When using GOMS, for instance, the designer does not need a detailed design description, since its action sequence is constructed in relation to the user's goals. Cognitive walkthroughs are therefore valuable, but only if performed quickly.

2.5.3 Choosing techniques

It is clear that neither guidelines nor cognitive walkthroughs can address all the design issues that may constitute usability problems. So designers and evaluators must decide which techniques to be used, based on time, money and expertise. It has already being demonstrated that there are significant benefits when usability is addressed, and problems eliminated. What is clear is that usability-engineering methods differ in their scope and application. The following table by Lewis and Wharton demonstrates differences between usability engineering and evaluation techniques.

<table>
<thead>
<tr>
<th>ABILITY</th>
<th>Cognitive Walkthrough</th>
<th>Heuristic Evaluation</th>
<th>GOMS</th>
<th>User Testing</th>
<th>Thinking Aloud</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test users</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Task specific</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Traces correct paths</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Maybe</td>
<td>Maybe</td>
</tr>
<tr>
<td>Assigns reasons for errors</td>
<td>Yes</td>
<td>Maybe</td>
<td>No</td>
<td>Maybe</td>
<td>Yes</td>
</tr>
<tr>
<td>Analyses user mental processes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Maybe</td>
</tr>
<tr>
<td>Estimates learning time</td>
<td>Maybe</td>
<td>Maybe</td>
<td>Yes</td>
<td>Maybe</td>
<td>Maybe</td>
</tr>
<tr>
<td>Estimates performance time</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Maybe</td>
</tr>
</tbody>
</table>

Table 2-3 The space of interface evaluation methods (Lewis and Wharton, 1997: 719)

But can any of these techniques guarantee that an application will be more usable if they are applied? This is a complicated question, and worthy of further study. But, at this juncture, it is necessary to ask what usability is in a practical sense. How does usability manifest itself and how do aspects of a design contribute to usability or yield usability problems? Most importantly, if the factors that contribute to usability can be identified, will it be possible to anticipate usability without user testing? The next
section is a comprehensive analysis of product quality as it relates to usability and the attributes associated with good interface design.

2.6 Properties, attributes, principles and quality

Usability is etymologically the “ability to be used”. However usability is typically defined in relation to underlying factors such as effectiveness, efficiency and satisfaction as in the ISO definitions (see Bevan 2001), or more comprehensive factors (as in Nielsen’s 1994 heuristics). But from an overall perspective usability is a critical factor in determining a ‘good’ user interface design, from a ‘bad’ interface design. This comparison is ambiguous without first examining how to distinguish ‘good’ design from ‘bad’ design. Since Cooper (1995) argues that there is a strong need for ensuring that consistent and appropriate terminology is used when discussing usability design, this section provides concise definitions for quality related terms.

The definitions adopted for a principle, heuristic, property and attribute are as follows:

**Principle:** “a rule or code of conduct”
(from [http://www.m-w.com/cgi-bin/dictionary: 2003](http://www.m-w.com/cgi-bin/dictionary: 2003))

**Heuristic** “Recognised usability principles” (Nielsen 2004: 45-9) or rules of thumb “derived from more extensive collections of interface guidelines.” (Cockton, Lavery and Woolrych 2003: 1122).

**Property:** Either the attributes of an entity type or “thing” (e.g. “the weight” of an object) or a relationship between entities such as “cars owned”. (Parsons and Cole, 2004).

**Attribute:** “an attribute is a given fact, which has been ascribed, imputed, assigned or otherwise given ” (van den Heuvel, 1996).

2.6.1 Quality and product quality

Quality is best described as a ‘desirable property’ meaning a positive aspect of a product (Dromey 1998). According to Price (1969) quality is entirely dependent on our ability to compare, characterise and evaluate the differences between different entities. These entities include sets of requirements, design guidelines and the product or products under investigation. When this type of information is carefully obtained
through observation and analysis, it becomes what Wilson (1967) describes as ‘true knowledge.’ This is why Price warns us to avoid value statements and expert opinions, which undermine our ability to reliably identify quality.

Quality is often described as ‘good’ ‘high’ and ‘best’ quality when a product is more desirable, and ‘bad’ ‘low’ and ‘poor’ when it is not. Thus reliably determining quality, or lack thereof, is dependant on the needs and requirements of a user. Wilson (1967) demonstrates this point by providing a simple example, with a horse as the product in question. In Wilson’s example, a farmer and a jockey describe a horse as good and bad, based on a set of different and often conflicting requirements. The farmer wants a strong horse for farm related work, whereas the jockey wants a fast horse for winning races. In each instance, the two people are determining the quality of a horse against a set of relevant characteristics, many of which are different to the other. A jockey for instance, is unlikely to ride an oversized Clydesdale, which, due to its lack of speed, is of poor quality as a racehorse. However, its strength and weight make it an excellent choice of horse for pulling a plough. Thus quality is entirely dependent on what characteristics are most important to a user.

The relationship between the characteristics of a product and its quality is demonstrated by Price’s (1969) definition of quality as “a recurrent feature of the world” and entities within “which presents itself in individual objects or events taken singly. Redness or bulbiness or squeakiness are examples.” This definition does not address the fact, as demonstrated by Wilson’s previous example, that some characteristics may be undesirable in a given context. This is why Dromey (1998:3) describes quality as a set of attributes through which “different interest groups express their needs.” These interest groups represent users, buyers, designers, users and anyone else, who has some stake in the given product. Therefore, the definition of quality adopted by this study is defined as follows:

**Quality:** A set of desirable properties, which expresses the needs and requirements of “different interest groups” or stakeholders. Quality represents conformance to these requirements.

(Dromey 1998: 3 and Juran cited in Firquin 1992:3)
According to Dromey (1998), Quality may be broken into a set of properties described as quality attributes. They are defined as follows:

**Quality Attributes (Qualities):** "desirable properties" (Dromey, 1998: 3)

The above definition defines quality in a very broad sense. However, the practical application of quality involves the design or application of a specific product. Garvin (1994) calls this 'product quality', which is described as follows:

**Product Quality** – Quality specific to the product in question based on one or more different quality dimensions including attributes, user needs, consumer preferences, and the value of the product in question. (Garvin 1984).

There are two problems with addressing all the properties or characteristics that contribute to product quality. The first problem is that some characteristics can be in opposition to one another (Galitz 1997) as in Wilson’s example where the characteristics (strength and speed) of the product (the horse) were not entirely compatible. Galitz (1997:45) argues that design is "based on a series of trade-offs balancing often conflicting design principles." The second problem is that the circumstances of a given project may also prevent the designers from addressing all the quality producing attributes (Garvin 1984). Both these problems have a significant impact on the application of product quality. Since the designers will not be able to address every single characteristic associated with quality, they must often make trade-offs between them.

**Trade-off:** The sacrifice or weakening of a quality producing principle or attribute because of either another conflicting quality producing principle or attribute, or project constraints.

Examples of trade-offs in usability design may include making a user interface simpler, but less efficient than a more complex design.
2.6.2 Linking quality to quality attributes

So is quality able to be concretely determined by looking at underlying attributes? Dromey (1998) proposes a software quality model (SQM) for identifying and preventing quality related problems in software. The SQM involves linking a model of quality to a model of the system. This approach is slightly similar with the system and design models proposed by Norman, where different models of the product exist between the designer and the engineer. The purpose of the SQM is to compare quality related properties (i.e. attributes) to physical aspects of the design (i.e. the code) in order to find defects in order to prevent errors.

SQM is criticised by Lauesen and Younessi (1998) who argue that not every defect can be traced to a violation of a tangible quality carrying property. They also argue that only 45 percent of the defects can be found in the underlying code. But the SQM is not limited strictly to the code, since Dromey acknowledges that usability is a major factor in determining the quality of a system. Based on these arguments it is possible to identify three principles on software quality and related problems. These are as follows:

1. Code alone is insufficient to determine every factor that may cause a defect.
2. That some, but not all, quality-carrying properties are evident in the software.
3. The remaining defects must be evident elsewhere.

The above argument demonstrates that code alone is unable to determine the quality of software and that usability as an important quality in software design. So is usability, the product of underlying attributes, identifiable in the design? The next section discusses the quality attributes associated with usability and their value in preventing usability problems and predicting usability.

2.7 Usability and its attributes

If usability is a form of quality and quality is itself the product of underlying attributes, then it follows that there are quality related attributes specific to usability. In order to predict usability, these usability attributes must be somehow identified, before deployment, during the design or redesign stages of an application. What are
these attributes and how can they be identified? This section analyses the opinions and guidelines associated with usability design and how to find usability attributes.

The first place to look for the attributes that underlie usability is within the HCI principles associated with user interface design and usability. These principles are a top-level set of design recommendations espoused by various HCI experts. These rules and recommendations are intended to assist designers in improving the usability of their applications. This study looks at three sets of these principles, provided by Shneiderman (1998), Nielsen (1994) and Galitz (1997).

Table 2-4 Shneiderman’s Principles

<table>
<thead>
<tr>
<th></th>
<th>Shneiderman’s Principles</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Strive for consistency</td>
</tr>
<tr>
<td>2.</td>
<td>Enable frequent users to use shortcuts</td>
</tr>
<tr>
<td>3.</td>
<td>Offer informative feedback</td>
</tr>
<tr>
<td>4.</td>
<td>Design dialogs to yield closure</td>
</tr>
<tr>
<td>5.</td>
<td>Offer error prevention and simple error handling</td>
</tr>
<tr>
<td>6.</td>
<td>Permit easy reversal of actions</td>
</tr>
<tr>
<td>7.</td>
<td>Support internal locus of control</td>
</tr>
<tr>
<td>8.</td>
<td>Reduce short-term memory load</td>
</tr>
</tbody>
</table>

Adapted from Shneiderman (1998:74 - 75)

One of the most prominent and regularly updated texts on HCI and user interface design is Shneiderman’s (1998) text on user interface design. Shneiderman’s (1998) set of design principles, contained in Table 2-4, is very general. This is why he argues that these principles will need to be adapted to fit a specific environment. He also acknowledges that these principles are by no means comprehensive, since more principles will be needed for addressing error prevention and user diversity.

While these guidelines seem credible, they are not without their critics. Cooper (1995) for instance, argues that consistency is not a valid design principle. His reason is that consistency limits innovation. However Galitz (1997) also believes consistency to be an essential aspect of design, though acknowledges it may need to be traded off.

Another set of principles can be found in Nielsen’s ten usability heuristics in Table 2-5 (Nielsen 1993). There is little difference between heuristics and principles. Heuristics and principles are both rules of thumb, but according to earlier definitions, heuristics

Table 2-5: The Ten Usability Heuristics

<table>
<thead>
<tr>
<th></th>
<th>The Ten Usability Heuristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Strive for consistency</td>
</tr>
<tr>
<td>2.</td>
<td>Enable frequent users to use shortcuts</td>
</tr>
<tr>
<td>3.</td>
<td>Offer Informative Feedback</td>
</tr>
<tr>
<td>4.</td>
<td>Design dialogs to yield closure</td>
</tr>
<tr>
<td>5.</td>
<td>Offer error prevention and simple error handling</td>
</tr>
<tr>
<td>6.</td>
<td>Permit easy reversal of actions</td>
</tr>
<tr>
<td>7.</td>
<td>Support internal locus of control</td>
</tr>
<tr>
<td>8.</td>
<td>Reduce short-term memory load</td>
</tr>
</tbody>
</table>

From Nielsen (1994)
may be educated guesses with no guarantee of success. But for all intents and purposes, when compared with Shneiderman’s principles, Nielsen’s heuristics may also be considered principles.

Nielsen is a strong usability advocate who acknowledges that user testing is not always viable, due to time, money or expertise. In order to provide what Nielsen calls a ‘better than nothing’ approach, he analysed a large number of usability problems to produce his ten usability heuristics. While Nielsen and Shneiderman’s principles both advocate error prevention, error recovery, user control, consistency, help and documentation and feedback, Nielsen’s guidelines also encourage simplicity (minimalism), aesthetic design and familiarity. Heuristic nine is equivalent to Shneiderman’s eighth principle, since both seek to prevent memory load.

Another, more extensive, set of principles is provided by Galitz who identifies 18 different principles (Table 2-6) related to user interface design. Surprisingly, Galitz proposes trade-offs as a principle to acknowledge that some of these principles are likely to conflict with each other.

<table>
<thead>
<tr>
<th>Aesthetically -Pleasing</th>
<th>Control</th>
<th>Predictability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clarity</td>
<td>Directness</td>
<td>Recovery</td>
</tr>
<tr>
<td>Compatibility</td>
<td>Efficiency</td>
<td>Responsiveness</td>
</tr>
<tr>
<td>Comprehensibility</td>
<td>Familiarity</td>
<td>Simplicity</td>
</tr>
<tr>
<td>Configurability</td>
<td>Flexibility</td>
<td>Transparency</td>
</tr>
<tr>
<td>Consistency</td>
<td>Forgiveness</td>
<td>Trade-Offs</td>
</tr>
</tbody>
</table>

Table 2-6: User Interface Principles
Adapted from Galitz (1997: 36 – 45)

Based on the previous principles of Galitz (1997), Shneiderman (1998) and Nielsen (1994), it is possible to categorise their principles into five different aspects of user interface design.

These categories are as follows:

- Error handling: prevention and recovery
- Interaction and control
- Learning and understanding
- Accommodating user diversity
- Subjective attributes.
2.7.1 Attributes for error handling

Before discussing how to handle errors, it is best to establish what an error actually is. Reason describes errors as

"those occasions in which a planned sequence of mental or physical activities fail to achieve its intended outcome, [and] when these failures cannot be attributed to the intervention of some chance agency"  
(Reason 1990: 9)

However a more concise definition is provided by Wood, who defines errors as "incorrect actions performed by actions while performing a task." (Wood, 2000: 12).

Wood classifies errors into four categories:

1. Action Error: Incorrect physical action
2. Slip and Lapse: Failure to execute a stage in an action sequence
3. Mistakes: failure in the selection of an objective or means to achieve it.
4. Accidents: an undesired unexpected event

For the purposes of this study, Wood's (2000) definition applies. Further discussion on errors and how they apply to GOMS may be found in the next chapter.

Ultimately, humans make mistakes. This is why Shneiderman (1998:75) stresses that interfaces should "Offer error prevention and simple error handling." It also explains why error handling is part of Shneiderman, Nielsen and Galitz's principles for user interface and usability design. Since errors significantly impact on the usability of a UI, a good designer will address error handling. Error handling is a two-fold process. Using Galitz's terminology, error handling involves both preventing human error (Forgiveness), and recovering from an error (Recovery) when it occurs.

Forgiveness

Forgiveness is the acceptance and prevention of human error. This includes automatically formatting text when a number or sequence of characters is incorrect. Related guidelines include Smith and Mosier (1986) section 4.3 (Interpreting Misspelled Commands), and Sections 8 2 of NASA (1996)
Error recovery

Error recovery is unlike forgiveness where, instead of preventing errors, recovery corresponds to what happens after a user makes an error. Shneiderman’s principle is to allow “easy reversal of actions” (Shneiderman, 1998: 75). According to Galitz, recovery is determined by whether the system permits commands to be cancelled or reversed and for the user to return to a given point if an error occurs.

In practical terms, recovery essentially means a design has error correction (see section backup capabilities in Schumacher 1992: 3.1.2.13) and the provision of ‘UNDO’ functions (NASA 1996: 2.4.1).

2.7.2 Attributes for interaction and control

The user interface is designed to facilitate the interaction between user and computer. There are a number of different properties that concern the functional aspects of the interface and their affect on user interaction.

Responsiveness

According to Nielsen (1993:139), “user interfaces should have a high degree of responsiveness.” Responsiveness is the degree to which the system rapidly responds to user actions, through visual, textual or auditory responses (Galitz, 1997). Thus, Responsiveness is concerned with how rapidly and effectively feedback is provided by the system. Examples include providing progress bars to let users know how long it will take to finish an action and keeping waiting time to a minimum. According to Shneiderman (1998), while the response time is important, other factors related to the task and the user’s characteristics will also determine how quickly responses are made. However, Responsiveness is the minimisation of response time and the effective provision of feedback.

Design guidelines that apply to Responsiveness may be found in Smith and Mosier (1986: section 3.1/2), “Feedback Prompts and Cues” in Schumacher (1992: 202). The Goddard centre guidelines also contain numerous references including sections 4.1.6.3 “feedback for each keyed entry within 0.02 seconds”, 2.2 “Provide Informative Feedback” and 11.5 “Allow viewers to immediately see the results of their actions”
Controllability

Interaction is a relationship between the system and user whereby the user interface is shared and controlled by both of them. This concept is called the ‘locus of control’ (see Borsook 1991) with the user’s control over the system defined as ‘an internal locus’. The system’s control is the external locus, in the sense that the system is controlling the interaction. Some systems demonstrate a high external rate of control where the user is unable to do much. Others have a high internal rate of control where the user controls most of the interaction.

Generally speaking, HCI analysts recommend building user interfaces and systems that support an internal locus of control. Nielsen (1993) uses a study by Kay (1989) to demonstrate that where users have a high degree of control, they have a greater positive attitude towards the interface than those who don’t. In order to support the user’s locus of control, Shneiderman (1998) recommends removing ‘surprising’ and unpredictable system actions, as well as reducing the difficulty of performing a task. Galitz (1997) suggests that controllability is achieved when “a person working at his or her own pace, is able to determine what to do and how to do it.” (Galitz, 1997: 40)

(Galitz 1997: 40) suggests four practical ways of doing this.

- “The user must control the interaction.
- Actions should result from explicit user requests.
- Actions should be capable of interruption or termination
- The user should never be interrupted for errors”

It is important to note that the final guideline (that the error never be interrupted for errors) is at odds with situations where safety is paramount. Such an example can be found in Glendon and Stanton (2000) where control in an energy distribution company is surrendered to a safety management system whenever safety issues arise and then passed back when resolved. Situations such as this demonstrate that principles such as controllability must be traded off, as Galitz (1997) suggests, when circumstances warrant.
Control is achieved when a user has a satisfactory level of control over the interface. An example of good controllability includes the ability to skip an animation sequence, without being forced to watch it in its entirety (refer to Halgren, Fernandes and Thomas 1995). Practical guidelines are contained in both the Schumacher Ameritech guidelines (1992: 1), and the NASA (1996) Sequence Control Guidelines.

Directness

Directness is an attribute related to the functionality of direct manipulation systems (Galitz, 1997). Seeing what needs to be done and then performing it, both directly and visually, usually via use of a mouse, is now so much a part of the GUI environment, that to find a new application that does not currently use direct manipulation is rare indeed. According to Dillon and Leonard (1995: 48-49) direct manipulation refers to the “simplification of the interface that occurs when the users can manipulate objects directly without having to memorize and use abstract commands.”

Galitz (1997: 40) directs designers to provide “direct and intuitive ways to accomplish tasks”. Providing a direct sequence of actions to complete a task is also recommended by Shneiderman who outlines this concept as follows:

“By pointing at visual representations of objects and actions, users can carry out tasks rapidly and observe the results immediately”

(Shneiderman, 1998:72)

Directness as an attribute should be used cautiously. The most ‘direct way’ of entering text may not be the ‘best’ or simplest way. For instance, Nielsen argues that direct manipulation is not suitable for disabled users. As far as practical guidelines go, direct manipulation is so deeply embedded in our graphical UI based operating systems, that only older guidelines such as Smith and Mosier’s (1986: 276) deal with this issue specifically.

Efficiency

Efficiency is the minimisation of the user’s efforts in relation to productivity. Ortega, Perez and Rojas (2003:13) describe it as the capacity of software to perform, in relation to the conditions and resources available. Cooper (1995) suggests that Efficiency is ill defined and it is hard to determine what user interface efficiency
relates to, since it could apply to several factors including code, the ease of learning and the ease of use. Despite this, there are two clear aspects to Efficiency. These are the minimisation of the user’s effort and the efficient execution of system processes. Galitz (1997) recommends minimising user actions (e.g. eye-strain and hand movements) and ensuring the smooth transition between controls (e.g. windows). Efficiency clearly relates to the pace of interaction and the minimisation of time and effort.

Efficiency can be best described as the minimisation of effort to support increased productivity. Though this is not addressed specifically by Shneiderman, he does encourage designers to “Enable frequent users to use shortcuts” and improve the “pace of interaction” (Shneiderman, 1998: 74). Efficiency involves the minimisation of physical effort and wasted time. This is an important part of interface design. For instance, Nielsen (1993:26) includes Efficiency as one of the five components of usability and describes it as follows: “The system should be efficient to use, so that once the user has learned the system, a high level of productivity is possible.” But what are the practical implications?

Rules related to efficiency may be found in a number of documents. These include, Minimal User Actions (refer to Smith and Mosier, 1986, 207) as well as accelerators in Schumacher (1996:76) and the minimisation of data entry in NASA (1996).

2.7.3 Attributes related to learning and understanding

While control and error handling may well be self evident in the design, predicting how user’s think and behave is far more difficult. Those who advocate user testing as the sole technique for determining user interface quality would probably say ‘case closed’. Those who advocate cognitive modelling would strongly disagree. There are even those who suggest that in the future, there may well be less need for user testing, since simulated users will be so advanced. Whether or not systems will ever be able to accurately predict human behaviour is still in dispute. However, there are several identifiable attributes that support learning and understanding.
Learnability

Learnability is the way in which an interface supports learning. There are a number of factors related to this attribute. These include:

- Ease of learning
- Time to learn
- Structure of information (Mayhew, 1992)
- Provision and structure of learning support (e.g. tutorials, on-line help and wizards).

Nielsen (1993: 30-31) argues that Learnability is the most “fundamental” of attributes related to usability because it is the initial factor in determining the success or otherwise of the user’s use of a system.

There are strong claims that the Natural GOMS Method discussed in the next chapter can actually predict learning time, based on pre-determined values in a similar way to how execution time is determined (see Gong and Kieras 1994, Kieras 1997). This method is also claimed to provide a better structure for on-line documentation and help (Gong and Elkerton 1990). An additional set of guidelines for helping the user can be found in Section 2.8 of the NASA (1996) user interface guidelines, which provides suggestions on how the system can be designed to better assist users.

Predictability

This attribute is closely related to Learnability. Predictability is a measure of how well the interface supports the logical flow of interaction. According to Galitz, the “user should be able to anticipate the natural progression of each task”. Thus, Predictability can be summarised as supporting the user’s ability to determine what to do next. The logical flow of interaction is reliant on well-defined processes. This attribute is well supported by the GOMS techniques, which are intended to structure tasks in relation to the user’s goals. Other task analysis techniques such as hierarchical task analysis also assist designers in determining logical and efficient ways of performing a task. Galitz (1997) suggests providing cues, meeting user expectations and making visual elements, distinct and recognisable. He also stresses that the Consistency of the user interface, another quality attribute, has a strong influence on Predictability.
Smith and Mosier (1986: 189) suggest that the system’s response should be compatible with the user’s expectations, such that any change in state “is displayed in an expected or natural form.” They also stress that designers should not assume that a user’s expectations will match their own, and if necessary the users expectations can be gleaned through interview, questionnaire or user testing. Though they also argue that being consistent across the design can be used to ensure a valid response. Predictability is therefore determined by whether the interaction is both logical and expected.

**Simplicity**

The reduction of complexity is one of the fundamental goals of HCI. Nielsen (1993: 115) strongly agrees. He argues:

> “User interfaces should be simplified as much as possible, since every additional feature or item of information on a screen is one more thing to learn, one more thing to possibly misunderstand, and one more thing to search through when looking for the thing you want.” (Nielsen, 1993: 115).

Nielsen calls this principle “less is more.” In a similar vein Cooper (1995) suggests the “Occam’s Razor” principle that “simpler is better”. A practical example is provided by Cooper, where the resizing features for a calculator are left out, since they were unnecessary and only make the program more complex. According to Shneiderman, the reason for Simplicity is that humans have a limited memory capacity. This is demonstrated in Shneiderman’s (1998: 75) eighth principle, “reduce short term memory load.” He encourages keeping the display simple and provide sufficient training in relation to both mnemonics and sequences of actions to perform a task.

Simplicity is simply the reduction of complexity across all aspects of the interface. Some practical guidelines for Simplicity include ‘Minimal Punctuation’ (Smith and Mosier, 1986: 220) and “Only Necessary Information Displayed” (Smith and Mosier, 1986: 251).
Transparency

Transparency involves keeping the system ‘invisible’ or ‘transparent’ by hiding unnecessary detail. It is the concealment of the workings and technical details of the system to facilitate interaction. In a transparent user interface users should not need to be aware of the system mechanics in order to use the software. In other words, the workings should be kept invisible (Galitz, 1997, Cooper 1995 and Shneiderman 1998). Cooper (1995) provides the analogy of reading a book where the reader is undisturbed by the techniques used to write it, such as the printing or binding.

As far as the user is concerned, the inner workings of the tool should disappear allowing the user to apply their intellect to the task at hand (refer to Rutkowski cited in Shneiderman, 1998: 202). Thus Transparency is closely related to Simplicity since it prevents unnecessary complexity of information, though Transparency serves a more specific purpose.

Familiarity

Familiarity is an interesting quality, since it relies on the user’s previous experience. Familiarity is closely tied to three factors:

1. Familiar concepts and language
2. A natural interface, which mimics the patterns of the users’ behaviour
3. ‘Real world metaphors’ (Galitz, 1997: 41).

The concept of using ‘real-world metaphors’ is demonstrated by operating systems, which use the “desktop” metaphor along with similar real world concepts (e.g. “files” the “trashcan” and “windows”). It is intended to make the user interface ‘more intuitive’. However, some HCI experts advise the cautious use of metaphors (see Cooper 1995, Nielsen 1993). Cooper (1995: 54) describes the positive use of metaphors as “an insidious” myth. Nielsen advises that designers should not imply “too much” when applying a metaphor, since there will be significant differences in functionality between how the metaphor is realised in the software design, and its ‘real-world’ origin.

Familiarity should not be measured in relation to the use of metaphor, but in making a user interface more natural and providing familiar words and concepts. NASA
(1996) guidelines demonstrate two practical aspects of Familiarity as in “3.15/6 Familiar Wording” which states: “where possible use wording that is familiar to users”.

Consistency

According to some, providing Consistency is a valuable design principle. Lewis (1989) cited in Nielsen (1993) suggests that users will be more confident and willing to learn if the same action has the same effect every time. Shneiderman (1998) suggests that Consistency is largely responsible for the success of systems. In contrast, Cooper believes that Consistency can be abused. Cooper provides an example where representing two different types of button in a similar way camouflages their distinct and different functionality. Consistency may conflict with Predictability and Compatibility (see Shneiderman, 1998: 13) and new or innovative functionality. This is why some Consistency is likely to be traded off in favour of other attributes.

Lewis and Rieman (1994: 110) propose a concept called the ‘Intelligent Consistency Principle’. They argue that it is useful to borrow one part of a system and put it in another. However they warn that care must be taken to ensure these new parts of the interface do not clash with other existing parts, which are significantly different.

Rules for Consistency may be found in most of the guideline documents, with rules for consistent control formats in Smith and Mosier, 1986: 189, consistent pointer shapes in Schumacher (1996:85) and consistent interface characteristics in sections 3.5.3 and 2.11 of the Goddard Centre’s guidelines (NASA1996).

2.7.4 Attributes for user diversity

While the user interface may not be able to take into account all the individual differences of its users, it is possible to provide users with some choice as to how they use the system. This includes the ability to configure the system, the option to perform tasks quickly at a high level of expertise and easily at a lower level of expertise. User diversity also involves Accessibility for users with special needs. Accessibility is a significant and important part of Usability in relation to visually impaired users and other users with special needs. This is an important and valuable principal, but due to
the scope of this study is not covered in great detail. However Configurability and Flexibility relate to differences in user expertise (novice and expert) and preferences and are discussed in further detail here.

**Configurability**

Configurability is the degree to which the software allows the users to pick preferences and settings. These preferences may change the functionality and display of the application in obvious ways. Galitz (1997: 38) insists that Configurability involves the "easy configuration and reconfiguration of settings" and "the allowance for personal preferences." With the introduction of the Internet, Configurability has significant security issues especially in relation to passwords and personal information.

In relation to software, Configurability is generally a good thing, in the sense that the user may structure the environment to suit their needs and preferences. However caution must be taken that these settings are easily reversible to default settings, in case the user makes a mistake. This is why many programs offer a means of returning a system to its 'default' settings. Practical guidelines for Configurability may be found in NASA's (1996) guidelines in:

- User Assigned Command Names (219)
- The Definition of Macro Commands (235)
- Alarm Definition by Users (245).

**Flexibility**

Unlike Configurability, Flexibility is performance orientated. According to Galitz (1997) Flexibility is a system’s capability to respond to individual differences between users. This is best demonstrated in the differences between novice and user behaviour. For instance, Nielsen (1993) suggests that “accelerators” be provided for expert users so that in his example, Flexibility means providing alternate ways of interaction for different levels of expertise. Flexibility is about short cuts for expert users and easy or informative techniques for novice users.
Practical references for Flexibility include 3.16/5 (Schumacher 1992: 225) and 3.0/3 (Schumacher 1996:187). Flexibility is also an important part of the Goddard Centre Guidelines section on sequence control (NASA: 1996, 5.5.1).

2.7.5 User dependent attributes

There are several attributes that do not fit comfortably into the previous two categories. These attributes tend to be more qualitative, though efforts have been made to quantify them.

User Satisfaction

Unfortunately, User Satisfaction cannot be directly determined without the ability to ask or observe users who have used the software (Olsen Jr and Halverson, 1988). Nielsen argues (1993) that an application should be designed to be “pleasant to use” to ensure that the user will “like” using it. While many of other attributes undoubtedly contribute to eventual satisfaction of the user, the only way to presently determine this is through actual user testing, or other methods that involve the user. Recent developments in eye tracking technology promise more accurate measures of satisfaction in user testing. However, satisfaction cannot yet be accurately predicted nor determined in the design and task analysis stages without user involvement.

Visual Appeal / Appearance and Layout

The quality of the visual display is not an entirely a subjective factor. For instance, it can be checked against guidelines and style guides with rules on colour, alignment and positioning. For example, Lewis and Rieman (1994) provide three practical principles for the design of the display.

- **The Visibility Reflects Usefulness Principle** – Hide infrequent controls, and make frequently used controls more obvious.
- **The Clustering Principle**: Put similar controls into visually separate blocks on the screen.
- **The Reduced Clutter Principle**: Don’t put too much on the screen.

(See Lewis and Rieman, 1994: 3.10 and 3.9)
Examples concerning colour and layout may be found in Schumacher (1992)'s guidelines in rule 4.3.1, and in Section 7 of the Goddard Centre's guidelines NASA (1996). Visual guidelines for web pages may be found in Ameritech's web guidelines (Detweiler and Omanson, 1996) in section 11.4. Other rules are available from Galitz (1997) who lists his rules under the term 'aesthetically pleasing.' There are also guidelines for how sound may be used, but these are not pertinent to this study.

2.7.6 Attributes not covered in the study.

There are other attributes not covered in the study. These include Accessibility, Security (which may also have a negative effect on usability), and Comprehensibility. Each of these attributes has an impact on the usability of a system and are extensive topics in their own right but are not covered in detail here. Figure 2-1 contains the attributes and categories discussed in this chapter.

![Figure 2-1 Proposed attributes contributing to usability](image)

2.7 Summary

This chapter has reviewed user interface quality, the quality of use (usability) and the properties that contribute to quality. The following chapter addresses user interface design and evaluation in greater detail, focusing on task analysis. Comparisons are made between various task analysis techniques and the GOMS and GSE modelling techniques are introduced. Most importantly though, it examines how the quality producing properties outlined in this chapter, integrate with the task analysis and evaluation techniques in the next. Thus, the next chapter takes the concepts and properties in this chapter and examines their practical implications.
CHAPTER 3
REVIEW OF THE LITERATURE:
TASK ANALYSIS, COGNITIVE ENGINEERING
AND SOFTWARE ENGINEERING

"It's one thing to meet the requirements of a contract and another thing
to build a good system. Are anybody's interests really served, if you
build a system that meets specs but is a failure?"

(Lewis and Rieman, 1994: Chapter 2:2)

3.1 The freedom of design

The previous chapter introduced several properties and attributes of quality that
are evident before implementation (see Table 3-1 and Figure 3-1). These
properties were proposed as a potential way to address the third requirement
specified in the NITRD's (2003) research needs. However these properties alone
are unable to fulfil the NITRD requirements for a "total system model." The aim
of this chapter is to determine a cognitive processing model and a total system
model that are compatible with the quality attributes proposed in the previous
chapter. Since both must be able to be used prior to deployment, it is important
also to discuss the early stages of design, particularly task and requirement
analysis where the modelling of the system begins. This chapter discusses
whether there is a viable means of meeting the NITRD requirements with existing
design techniques and specified usability properties.

![Diagram of the relationship of the NITRD models in the early design process]

This chapter is divided into three main sections. The first part discusses the pre-
deployment stages of design and, in particular, task and requirements analysis.
The second section discusses cognitive modelling and related techniques (e.g. GOMS) in order to find an effective means of modelling the user's behaviour. The final part of this chapter examines software-modelling techniques including UML (Unified Modelling Language) and GSE (Genetic Software Engineering). Since the cognitive and system models must address predictable usability properties, Table 3-1 summarises the attributes associated with Usability (identified in the previous chapter).

<table>
<thead>
<tr>
<th>Error Handling</th>
<th>Interaction and Control</th>
<th>Learning and Understanding</th>
<th>User Diversity</th>
<th>Subjective Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forgiveness Recovery</td>
<td>Responsiveness Controllability Directness Efficiency</td>
<td>Learnability Predictability Simplicity Transparency</td>
<td>Configurability Flexibility</td>
<td>User Satisfaction Visual Appeal</td>
</tr>
</tbody>
</table>

**Table 3-1: Possible predictive properties.**

### 3.2 Task analysis and the early stages of design

According to Rosson and Carroll (2002:24), designing a product without an understanding of the user's needs, technical requirements and alternate solutions, requires either blind luck or excellent intuition to obtain a good design outcome. Kieras (2004b: 46-2) also argues that if the "*initial requirements and system functions are poorly chosen, the rest of the development will probably fail to produce a usable product.*" This means that the designer must pay careful attention to the intended purpose of the application; what it is expected to do

To satisfactorily determine whether an application meets its purpose, the requirements specified in the early stages of design must be compared against the application. Perhaps the most important requirement is ensuring that the design meets the user's task needs or what they want to do with the product. This involves the analysis of their tasks and thus this stage of design is called task analysis. However there are other requirements, both technical and functional, which must also be addressed, and so another early phase of design is called requirements analysis. If usability attributes can be anticipated before deployment, it is likely that some, if not all, can be found in these crucial early stages. This section discusses task and requirements analysis, in order to establish
what is needed to adequately represent the user’s task requirements in a total system model.

3.2.1 What is task analysis?

According to Diaper (2002a), HCI is a field devoted to understanding user tasks and is therefore focused on task analysis. So what is task analysis? Diaper (2004) provides a very broad definition, which encompasses almost any system analysis technique, which incorporates people within the design. He states:

“Task analysis is the collective noun used in the field of ergonomics which includes HCI for all the methods of collecting, classifying and interpreting data on systems that include at least one person as a system component.” (Diaper, 2004: 14).

However, Kiersas provides a more practical definition, given that it includes the purpose of performing a task analysis. He defines it as:

“the process of understanding the user’s task thoroughly enough to help design a computer system that will effectively support users in doing the task.” (Kiersas 2004b: 46-1).

So task analysis is aimed at ensuring the system allows the user to do their jobs, or “doing work to achieve goals” (Diaper 2004: 6). To do this, the analyst must identify those features of the environment in which they work that constrain both their behaviour and the completion of their goals (Card et al 1983).

Since the designers must ensure that an application does what its users want it to do, they must design the behaviour of the system to facilitate the user’s work activities. To do this, task-analysis decomposes these work activities into a description of actions and their effects (Carey 2002). By analysing these activities, the analyst is able to obtain information on how the application is expected to function, and therefore allows designers to determine the functional requirements for a system (Scogings and Phillips: 2001).

However, there are other requirements that designers must also address, which may be more technical or are not directly associated with the user’s tasks. For
instance, two high-level task requirements for a word processing application are the ability to enter text and the ability to save documents. Without meeting these requirements the word processor is unable to complete its intended purpose. But in order to save documents the application must have access to the computer’s file access system, which is a technical function on which many task requirements depend. In the previous chapter, the quality attribute called Transparency (see section 2.4.3 Galitz 1997) involved hiding system details that the user should not need to know, but are essential to the operation and functionality of the system. Tasks therefore depend on requirements that may not be directly associated with the task in question (e.g. file access).

According to Alm (2003), an application must be tested against a formal representation of its requirements. But what kind of information should a designer be able to obtain from a formal task analysis? Jeffries (1997) believes that every possible task in a client’s domain (e.g. their work environment) must be enumerated. But some tasks are clearly more important to the user or client than others. Saving a document for instance, is likely to be more important than checking the grammar or inserting a table. This is why Dumas and Redish (1999) argue that task analysis should provide the three following pieces of information:

1. The objectives of each task
2. The importance of tasks in relation to others
3. Which tasks do other tasks depend on?

One method of determining the importance of tasks (number 2 above), suggested by Scogings and Phillips (2001) is to determine the frequency of a task. For instance, how many times a day is an end user expected to check the grammar in a word processed document? Thus, task analysis is a detailed analysis of the user’s activities, with the intention of determining the objectives, importance and dependencies between these tasks.

There is some confusion on how task analysis relates to requirements analysis. This is not surprising given that Redish and Wixon (2003) identify four different
perspectives on what task analysis actually does, ranging from encompassing the entire pre-design process, to a single post-design activity.

For instance, Rosson and Carroll (2003) believe that task analysis is one part of a much broader requirements technique. In contrast, Diaper (2002b) believes that task analysis and requirements analysis are essentially the same thing. Diaper implies that task analysis should identify all of the requirements for a design. This study adopts the view that a distinction should be made between the user’s tasks and other requirements. This is to prevent ignoring the user’s needs in favour of technical requirements and functionality. This reason is clarified by Dumas and Redish who state:

“Product designers work so closely with the product that they must make a concerted effort to keep from becoming totally immersed in the functionality. It is all too easy to forget that the product exists because human beings are trying to accomplish tasks. Task analysis refocuses attention on users, their tasks, and goals.” (Dumas and Redish, 1999: 44)

Thus, task analysis is focused almost entirely on the tasks that the client or user wants to perform. Such a position distinguishes the user’s tasks from other technical requirements.

Task analysis approaches are usually embedded within a larger requirements analysis process. For instance, Scenario Based Design (SBD) (Rosson and Carroll 2002, 2003) is a requirements analysis technique based on stories of current work practices. SBD is essentially a subjective representation of the workplace. People who have a ‘stake’ in the eventual design (e.g. workers, managers, customers) provide stories or ‘Scenarios’ which describe the current practices in the target workplace. These stories are analysed and later subjected to task analysis as part of the SBD process. Another requirements analysis technique called Cognitive Work Analysis or CWA (Vincente 1999), also includes task analysis as a distinct part of a larger requirements analysis. CWA and SBD both demonstrate that task analysis can be conducted as part of a broader requirements analysis process.
However, there are potential problems if the task analysis is not conducted with sufficient rigour. SBD, for example, may not provide a task analysis that is as formal (Benyon and Macauley, 2002), or as detailed as a designer may need (Rosson and Carroll, 2002). According to Diaper (2002c) this lack of formality subjects the SBD task analysis to bias, yet at the same time also allows the designer to more easily address contextual requirements (Turner and Turner 2001). SBD demonstrates that task analysis techniques differ significantly in terms of detail, formality and subjectivity. This means that for a designer to pick an appropriate task analysis technique, there are a number of criteria that should be assessed.

3.2.2 Assessing task analysis techniques

Jeffries claims task analysis is unable to handle 'unstructured' tasks where multiple ways of finishing the task exist. Jeffries' second criticism is that task analysis techniques can provide too much data, making the actual analysis difficult. Since some analysis techniques are criticised for not being sufficiently detailed, there are recognisable differences between different task analysis techniques (Rosson and Carroll, 2002).

So, how should a designer choose an appropriate task analysis technique? Lim (1996, 32-33) presents a number of issues that task analysis techniques may or may not address. These are described as follows:

- A task analysis technique may be dependent on an existing system.
- There may be too much analysis.
- The actual domain (e.g. the workplace) may not be appropriately specified.
- The outputs of a task may not be adequately represented (e.g. what is passed on from one task to the next as part of the workflow).
- The task may provide insufficient information for selecting and applying different methods to complete a task, this is essentially Jeffries 'unstructured task' criticism.
- The task analysis technique may be poorly developed.
• The task analysis technique may not be explicitly related to software engineering and system development models.

Lim’s last point, that a task analysis technique may not relate to development methods, raises the issue of compatibility between methods. Diaper (2004: 30) notes that because many task-analysis methods were “developed by researchers with a psychological background”, they do not integrate well with software engineering.

Task analysis is only one stage of the development process. The value of a task analysis method is largely dependent on its outcome: an effective design based model of the user’s tasks. If this model is unusable or difficult to use alongside the other stages of the development lifecycle, its value is greatly diminished, especially since adapting them at a later stage to make them more compatible with software engineering representations is difficult (Diaper 2004). This supports the NITRDs support for ‘total system models’, which integrate many aspects of the design into a single model of the system.

There are a number of practical aspects that must be considered when choosing a task analysis technique. Arnowitz et al (2000) suggest that a successful task analysis must be able to do the following:

• Address the needs of a team, with members from different disciplines.
  (e.g. software engineers, and usability experts.)
• Be quickly executed in commercial projects
• Provide an easily read overview of the task structure
• Provide the designers with assistance and guidance on how to implement dialogue design
• Supply a technique for comparing the task analysis against the concept for the design
• Support the creative aspects of user interface design
• Be flexible enough that the task analysis technique can be integrated with other methods.
Lim (1996) and Arnowitz et al (2000) demonstrate some of the complexities involved in choosing a task analysis technique. With so many different task analysis techniques available, it is likely that many do not address their recommendations. The next section documents some of the more common task analysis techniques and their advantages and limitations.

3.2.3 Task analysis techniques

There is a wide range of available task analysis techniques. Some like scenario-based design require extensive contact with the users of the system. Others like HTA and GOMS (see 3.2.4 and 3.3.2) follow a predictive approach by linking user goals to their underlying actions. Some of these techniques are widely used and others are not. The last section demonstrated a number of factors that may be used to choose a task analysis technique. In order to better understand how task analysis works, this section examines several common task analysis methods.

Structured interviews

One of the more direct ways, to begin a task analysis is simply to ask the relevant people what they want the application to do. The designers or analysts can conduct interviews for obtaining this information. The advantage of this technique, according to Shneiderman (1998), is that the interviewer is able to address any issue of concern. This is because the researcher can ask anything about the design, although they should be careful not to make assumptions, nor translate the subject's words into product jargon (Redish and Wixon 2003). The down side, according to Jeffries (1997), is that subjects may provide the 'official' way of doing a task and not the way they actually do it in practice. According to Dix, Ramduny-Ellis and Wilkinson (2004:390) interviews can also be problematic because how people describe their actions, "are frequently at odds with what they actually do". An additional danger is that the subjects may 'contaminate' the analysis if they know too much about any changes that are proposed. Jeffries proposes that these questions can be allocated to a separate session. Perhaps the most significant problem is that interviews can be both costly and time-consuming (Shneiderman 1998).
It is important not to mistake these structured interviews (described in Kirwan and Ainsworth 1992) for ordinary interviews, which do not provide a formal means for transforming this data into user tasks. The ordinary interview is typically seen as just an evaluation technique (Shneiderman 1998, Blandford and Rugg 2002). However referring to the criteria in the previous section, structured interviews are neither fast, nor do they make any direct connection to software engineering.

Surveys

An alternative to interviewing subjects is to obtain information through surveys. These are not to be confused with interface surveys (Kirwan and Ainsworth 1992) which are a series of techniques the analyst uses to systematically analyse specific features of the interface. Jeffries (1997) and Shneiderman (1998) do not believe that surveys should be the sole means of task analysis and evaluation. Shneiderman (1998: 132) argues that surveys are “familiar, inexpensive and generally acceptable companion for usability tests and expert reviews.” He also suggests that surveys can provide a subjective impression of task objects and actions.

The disadvantage of surveys is that they may not determine all the important sub-tasks that may be performed, which is why Jeffries (1997) recommends using them with different techniques. Surveys may also be subject to bias, since not everyone who receives a survey will actually complete it (Shneiderman 1998).

Retrospectives and diaries

Retrospectives involve asking the observers questions after a session has finished. Another approach involves giving subjects diaries, in which the subjects are instructed to write down their thoughts at different intervals in a session. These methods rely on the efforts of the subjects and it is often difficult to get users to provide relevant information (Jeffries 1997).

Observations and shadowing

Observations consist of actually watching the users doing their work. This may involve one or many different users (Redish and Wixon 2003). Shadowing also involves following the user around their place of work and observing them as they
perform their activities. These methods are both labour intensive, but may be
more practical if the cost of the other methods is too high (Jeffries 1997).
However, according to Dix et al (2004:390), observation tends to fail in long-term
situations where activities are “sporadic” or “lockstep”, and when unusual events
can occur. Dix et al (2004) also argues that observation will be difficult for
processes that are spread across time and geography.

Contextual inquiry and participatory design

Contextual inquiries are a range of different techniques that target the broader
context of a task. Participatory design involves making the users co-designer.
Jeffries (1997) warns that the analysis should not be biased to a particular issue.

Other task analysis techniques

There are a number of other task analysis techniques that are worth investigating
Like all task analysis techniques, they have advantages and disadvantages.
Arnowitz et al (2000) for instance, proposes a technique called task analysis
maps. Based on their own admission, this technique is unable to represent
complex systems without inventing additional diagrams. The interesting aspect of
their diagrams is that they use colour to represent the dependencies between
objects. However, the incompleteness of the method, in regards to modelling,
suggests this approach is best applied to small systems.

Palanque and Bastide (1997) use a technique that involves a more formal notation
for modelling tasks. This, they argue, allows the model to be both mathematically
validated and able to formally model ‘conciseness’, ‘consistency’ and lack of
ambiguity. Palanque and Bastide use Petri nets (a type of process diagram) in
conjunction with an object orientated system model to establish a more complete
formal design framework. However, they admit that using Petri nets is likely to
make the model ‘hard to build, understand, and modify’ (Palanque and Bastide
approaches as not being able to represent high-level interaction or contextual
factors.
There are several recent task analysis techniques still under development. For instance, Scogings and Phillips (2001) describe a method called ‘Lean Cuisine +’ a technique which uses UML use cases and UML class diagrams along with a task action sequence to represent user behaviour (the UML framework is discussed later in this chapter). They note that the task-action sequences often overlap causing problems with the representation. Another recent technique called PACT (People, Activities, Context, Technologies) uses scenarios much like Rosson and Carroll’s SBD approach (3.2.1). These scenarios are analysed in terms of actions, objects and activities, ultimately merging objects and actions into a single entity (Benyon and Macauley 2002), on which to design the application.

3.2.4 Hierarchical Task Analysis HTA

One of the most popular task analysis methods is Hierarchical Task Analysis or HTA (Diaper 2004). HTA is a means of describing tasks as a hierarchy of goals and subgoals. While HTAs typically consist of hierarchical task description diagrams such as that in Figure 3.2, both Shepard (2001) and Annett (2004) are careful not to limit HTA to the diagrams produced by the method, but argue that the process for analysing the tasks is also a critical part of HTA also. Kieras (2004b) argues that since other task analysis methods present data hierarchically HTA would be better called procedural hierarchical task analysis, given its focus on a procedural representation of the task. However the best way to explain HTA is to break it down into its components.

HTA consists of four key components: goals, operations, tasks and plans. Tasks can be defined, as whatever work has to be done. Shepherd (2001: 22) states tasks are “a problem to be solved or a challenge to be met” and consist of a system goal, resources to be used, and constraints on those resources. According to Kieras (2004b), HTA tasks are a combination of a goal and the context (i.e. conditions) in which it happens. The next HTA components, Goals are a desired state of affairs (what a operator wants to happen). The third components, Operations, are the actual activities that are carried out in order to complete a goal, and are the “fundamental unit of analysis” in an HTA model (Annett 2004). Annett (2000) also argues that operations should not be described as “primitives” since they themselves can be broken into sub-operations. The last main
component, Plans, specify the rules governing the order of operations to be
carried out, including decisions that may need to be made, and any operations that
must be performed at the same time (Annett 2004).

According to Annett (2004) HTA is a seven step process with

1. Decide the purpose of the analysis
2. Get agreement between the stakeholders on the definition of task goals
   and criterion measures
3. Identify sources of task information and select means of data
   acquisition
4. Acquire data and draft decomposition table / diagram
5. Re-check validity of decomposition with stake-holders
6. Identify significant operations in light of purpose of analysis
7. Generate and, if possible test hypotheses concerning factors affecting
   learning and performance.


A core part of HTA (handled in step 4 of the above procedure) is the actual
production of diagrammatic representations of the tasks themselves. Standard
HTA diagrams can be represented in two different formats, as a functional

![Diagram: Hierarchical Task Analysis]

Figure 3.2: Hierarchical Task Analysis:
Shepherd 2001: 46

governing the order of procedure) are annotated on to the diagram.

The tabular format typically contains two columns (Shepherd 2001: 93). The first
column is the task description, essentially all the data in a standard HTA diagram
in point format and the second column contains notes, comments and any extra
information the analyst wishes to record.

Page 66
Despite its popularity HTA is not without its problems. Several studies have reported that HTA is difficult to learn and train in and apply effectively (Ainsworth and Marshall 1998 and Simon and Young 1998). Diaper (2004) argues that HTA is one of the hardest task analysis methods to apply effectively given that it is poorly specified and requires significant expertise. Also, according to Kiers, HTA methods have difficulty representing the flow of control, given that the structure of goals and sub goals has to be determined from the annotations for the sequence of actions. Though annotations exist in Figure 3.2, the boiling an egg procedure is fairly linear in its execution. Kirwan and Ainsworth (1992) also note that HTA requires a fair bit of time effort and cooperation in order to gather the information needed for the analysis.

It is easy to identify GOMS as being similar to HTA as Kirwin and Ainsworth (1992) maintain. Both for instance, share a hierarchical structure and rely on goals, sub goals and operations. However there are significant differences. First of all, according to Kiers (2004b), GOMS grew out of cognitive skill and problem solving, whereas HTA is based on common sense task decomposition. Indeed Shepherd (2001:72) notes that in HTA, cognitive operations can only be inferred by “considering the wider pattern of actions and operators intentions”.

A second issue according to Kiers (2004b) is that unlike GOMS models HTA has difficulty representing complex flow of control. This is made apparent by the fact that rules for the flow of control must be added as annotations to the diagram. In 3.2 the annotations are quite simple due to the linear nature of the diagram, however in more complex sequences this problem becomes apparent. In GOMS models, however, these rules (represented as selection rules and methods) are integrated into the notation as a whole.

Thus while this study does not dismiss the benefits or widespread appeal of HCI as a task analysis technique, it is inclined to use GOMS, due to its more overt inclusion of cognitive principles. In order to make these reasons a little clearer it is important to discuss both cognitive modelling as it applies to task analysis, and the GOMS methodology itself.
3.3 Cognitive modelling, Model Based Evaluation and GOMS

The NITRD (2003) HCI research needs specify that a "total system approach" to design must be based on cognitive principles. One application of these principles involves building models of human cognition in order to understand and anticipate user behaviour. According to Olson and Olson (1990) the advantages of using these models to influence design include:

- Preventing an excess of items in working memory
- Choosing more user friendly design options
- Estimating task performance time
- Estimating Learning time and producing efficient training documents
- Determining error-prone or inefficient activities and guiding HCI research.

Building models of how a human will use a system, and then applying them to a design to obtain usability data is known as model based evaluation (Kieras 2003). Two of the most relevant current approaches to model based evaluation are cognitive architecture models, and GOMS. Since the aim of this study is to integrate cognitive principles in a "total system" approach this section discusses both cognitive architecture and GOMS and establishes why GOMS was chosen as the basis for the approach proposed in this study.

3.3.1 Cognitive Architectures

Cognitive architectures are models of human thought based on the hypothetical interaction between perceptual, cognitive and motor processes (Kieras 2003).

According to Byrne, a cognitive architecture can be defined as a

"a broad theory of human cognition based on a wide selection of human experimental data and implemented as a running computer simulation program." Byrne (2003: 98)

Cognitive architectures are models of human thought based on data gleaned from psychological and related research. Cognitive architectures owe a great deal to the from Model Human Processor (MHP) concept proposed by Card Moran and Newell (also the inventors of GOMS) back in 1983.
Perhaps the best way to think of the MHP is as a human information process system, structured in a system-like architecture, with its own components and parameters that indicate performance measures (Lee 1995). Green (1999:6) describes the MHP as "a computer representation of human thought", split across three memory systems and four processors (including perceptual, cognitive and motor). According to Green cycle times for each of the processors are assigned best case, typical and worst case values. However like other cognitive architectures the MHP is reasonably complex. Green for instance argues that the MHP may be "too detailed" than is necessary.

There are several recent cognitive architectures worth mentioning. The EPIC or Executive Process-Interactive Control (Kieras, Wood, and Meyer, 1997) approach uses many of the principles associated with the MHP model. To construct a model in EPIC an analyst applies production rules (for specifying the knowledge used in a task) and perceptual motor parameters. EPIC allows the analyst to specify processors, such as speech and eye movement, that operate simultaneously (Kieras and Myer 1998). This may involve simultaneously recognising an object on screen, deciding what word should be spoken and pressing a key (Byrne 2003). However Byrne (2003) acknowledges that EPIC is unable to adequately address learning mechanisms.

ACT-R or Adaptive Character of Thought theory (Anderson 1995) is an attempt to represent complex conditions by encoding knowledge into production rules and units of knowledge called chunks. Put simply ACT-R allows an analyst to describe cognitive activities into rules for that activity and the knowledge involved. Some of the many activities ACT-R has been used to analyse include language comprehension (Edmund 1997), acquiring skills (Anderson and Fincham 1994, Taatgen, N.A. 2005), and spatial reasoning (Harrison and Schunn 2002, Kelley, Lee and Wiley 2000, Schoppek, 2002b). ACT-R has also been used by Salvucci (2001) to address parallel behaviour such as driving a car and speaking on a mobile phone. Much like GOMS ACT-R can provide predefined execution times to model activities. However, Salvucci (2001) admits that GOMS is a faster means of building additional behavioural models. It must be noticed,
that St Amant Freed and Ritter (2004, 2005) have recently developed a G2 system for converting GOMSL models into ACT-R notation.

There are several reasons why GOMS was chosen as the cognitive model for this study. Firstly, according to Green (1999), GOMS is the most commonly applied approach for estimating task times. Byrne (2003) acknowledges its applicability as “broad”. Secondly, according to Olson and Olson (1990), GOMS can model and assist the design of a large range of different applications in being both more learnable and more usable. Thirdly, they are not as complex as cognitive architectures that aim to represent psychological functions in greater detail (Kieras 2003). This is demonstrated by the size difference involved in converting GOMS models to their ACT-R counterparts (St. Amant Freed and Ritter, 2005). Finally, GOMS’s hierarchical representations of behaviour make it compatible with software engineering (total system) models.

3.3.2 The Principles and Components of GOMS

GOMS or Goals Objects Methods Selections is a prominent cognitive analysis technique. The original GOMS model invented by Card et al (1983) is based on two important concepts called the rationality and problem space principles:

The Rationality Principle: proposes that a user undertakes a set of rational actions to reach particular goals. These actions are limited by the user’s knowledge and capabilities (see, for example, Card Moran and Newell, 1983 and John and Kieras, 1996:b).

The Problem Space Principle: “The rational activity in which people engage to solve a problem can be described in terms of a (1) set of states of knowledge, (2) operators for changing one state into another, (3) constraints on applying operators, and (4) control knowledge for deciding which operator to apply next.” (Card Moran and Newell, 1983)

So GOMS is based on the assertion that people behave rationally to meet their goals and this rational activity can be analysed in terms of operators and knowledge. If both principles hold true, it should be possible to formally represent anticipated user behaviour by using some form of predictive analysis technique.
Components of GOMS

GOMS uses four components to represent human behaviour:

1. **Goals** are “a symbolic structure that defines a state of affairs to be achieved and determines a set of possible methods by which it may be accomplished” (Card et al 1983: 144).

2. **Operators** “an action performed in service of a goal. These include cognitive, motor, and perceptual operators “John and Kieras (1996b: 323). Physical operators in a graphic user interface include button presses, and mouse movement (Hudson et al 1999).

3. **Methods** “are learned procedures that the user already has at performance time. They are not plans that are created during a task performance. They constitute one of the major ways that familiarity (skill) expresses itself.” (Card et al 1983: 145)

4. **Selection rules** “allow us to predict from knowledge of the task environment (in this case the number of lines to the target) which of several possible methods will be selected by the user in a particular instance.” (Card et al 1983: 146)

3.3.3 Defining GOMS

Broadly described GOMS is a quantitative technique focused on human performance (Ivory and Hearst 2001). However its best-known feature is its ability to predict execution times for the duration of tasks (Green 1999, Beard et al 1997). But GOMS is much more than that. It can model tasks (Diaper 2004) and be used for model-based evaluation of designs (Kieras 2003). Virzi (1997) argues that GOMS is not a usability inspection technique, since it cannot identify usability problems. However this is clearly not the case borne out in studies including Chuah, John and Pane (1994), Gong and Kieras (1994), Gray et al (1992), Gong (1993) and Lee (1995). However to better demonstrate what GOMS can do, this section discusses its role in task analysis and the prediction of execution times.
Analysing a task

According to Kieras (1994) a GOMS model is

"a description of the knowledge a user must have in order to carry out tasks on a device or system, it is a representation of the “how to do it” knowledge that is required by a system in order to get the intended tasks accomplished." (Kieras, 1994: 371).

Despite its focus on the successful completion of tasks, there is some argument in the literature as to whether GOMS should be classified as a task analysis technique. In fact Kieras (2004a) and Kieras et al (1995) imply that GOMS takes place after a task analysis. Others suggest differently. For instance, Van Setten et al (1997) argues that GOMS suitability as a task analysis technique has been known for some time and Kirwan and Ainsworth (1992) discuss GOMS in a similarly capacity to the popular task analysis technique HTA discussed earlier. However, in other cases GOMS is used after other task analysis techniques are used (e.g. McLeod and Sherwood-Jones, 1992). The answer to this problem is largely dependant on what a task analysis technique is considered to be.

Logically GOMS deals explicitly with tasks. According to Dumas and Redish (1999) each task has a goal, starting point, a set of actions and a stopping point and GOMS does address these elements. But GOMS also fits nicely into the following definition of task analysis:

"Task Analysis is hierarchical. That is, each job can be broken down into a set of functions or tasks. Each of these tasks can then be broken down further into subtasks; subtasks can often be broken down further and so on.” Dumas and Redish (1999: 41).

Based on this definition it is clear that GOMS (especially later variants) adhere to these requirements. But while GOMS does analyse the procedural nature of tasks, it does not specify procedures for gathering the task requirements themselves. This is pertinent given that Diaper (2002) makes no distinction between requirements analysis and task analysis; they are one and the same.

If a task analysis technique must comprise all the steps necessary for obtaining the overall goals, such as those used in scenario based design and the early stages of
HTA, then GOMS is not a ‘full’ task analysis technique. In fact, this fits in well with Diaper’s (2004) assertion that GOMS is a ‘partial’ task analysis method given it addresses the later stages of task analysis. Kieras (2004:85) argues that GOMS does not replace “the most critical process in designing a usable system, that of understanding the users situation, working context, and overall goals.” Indeed, contextual and system requirements are not addressed explicitly by a GOMS analysis since it concentrates on the structure of the tasks themselves (Rosson and Carroll 2002, Palanque and Bastide 1997).

For the purposes of this study GOMS will be labelled as a task analysis technique, while acknowledging it does not cover requirement gathering, the explicit analysis of the users context, or the gathering of the overall users goals. These are covered by comprehensive task analysis techniques such as HTA, which specify these steps as part of the procedure (as specified in Annette 2000). However, GOMS addresses other issues that are not offered by task analysis techniques like HTA and Scenario Based Design.

**Predicting execution time**

Because GOMS allows a task to be broken down into well-known time operators (such as pressing a key) it is possible to estimate the time a task will take by assigning pre-defined times to these operators. Time values have been proposed by Card et al (1983) Gray John and Atwood (1992) Olson an Olson (1990) and MacKenzie (1991). However, GOMS has been criticised for only addressing error-free performance (Chuah et al 1994, Grey et al 1992), with different levels of accuracy for each GOMS method (Baskin and John, 1998). Despite this, Kieras (2004a) maintains that GOMS is an excellent starting point for addressing error-free performance especially in light of Wood’s (2000) heuristics for examining errors. While the approach in this study does not address execution time, since other means are used to determine the efficiency of a task, there is no reason why such a value could not be calculated from the relevant NGOMSL models.

**3.3.4 The variants of GOMS**

One of GOMS advantages is that it can “grow to accommodate new findings” (Karat 1998). There are now four variants of GOMS; the original GOMS model,
the Keystroke Level Model, the Critical Path Method GOM model and the
Natural GOMS language model

The original Card, Moran and Newell model CMN-GOMS

The original GOMS model by Card, Moran and Newell (1983) model was
originally used to model human behaviour while manuscript editing (i.e. word
processing). Card et al wanted to know if it was possible to:

- “describe the behaviour of a user engaged in text-editing as the
  repeated application of a small set of basic information-processing
  operators”
- “predict the actual sequence of operators a person will use and the
time required to do any specific task” (Card et al 1983: 139–140)

By breaking down users’ goals into a small set of physical and mental actions,
they attempted to not only model behaviour, but also to predict it. To do this they
employed their Model Human Processor or MHP. The MHP divides human
behaviour into three sub-systems: cognitive, perceptual and motor. According to
Card et al each of these subsystems has it’s own memories and processors. These
are summarised by several parameters including memory decay rate and processor
cycle time. Card et al realise that the MHP does not represent behaviour in its
entirety. They admit this by saying:

“A model so simple does not, of course, do justice to the richness
and subtlety of the human mind. But it does help us to understand,
predict and even to calculate human-performance relevant to

The original GOMS model can be conducted at different levels of detail or
‘grains’ of analysis called the unit-task, functional, argument and keystroke
levels. The unit-task model represents the high level tasks, the functional level
breaks these down into sub-tasks, the argument level adds arguments (i.e.
parameters) to these sub-tasks and the keystroke level GOMS model specifies the
original keystrokes and operators.

The Keystroke Level Model (KLM)

The Keystroke Level Model is the lowest level GOMS technique. However, it
does not use selections goals or methods (Hudson et al, 1999: 96) but describes
behaviour as a sequence of information processing operators, the times for which may be added to calculate the total execution time for a task (Card et al 1983). Operators include pointing, clicking, moving a hand to a device and mental operations symbolised by P, K, H and M respectively. Though dependant on its pre-determined time values this model has been found to be up to 80% accurate (Toleman 1996). However KLM is criticised for not addressing error making behaviour Balter (2000) or parallel activities (Baber and Mellor 2001). Updated time estimations (and values for new activities) may be found in several papers including John and Newell 1998, Olson and Olson 1990 and Myung 2004. KLM provides a quantifiable sequence of atomic actions for simple action sequences, but is unsuitable for high-level or complex task analysis.

CPM-GOMS.
Designed to represent parallel processes, CPM-GOMS uses schedule charts to describe how activity occurs in parallel across the perceptual, motor and cognitive processes. The critical path is the "sequence of tasks that takes the longest and determines the total time for the entire task" (Grey et al 1992: 309) and is represented by a line that zigzags between the different operators. Because of its low level focus, it can be labour intensive to apply (Kieras, Wood and Myer 1997), especially manually (Vera, Howes, McKurdy and Lewis 2005), although a tool called APEX has been developed to speed the process John, Vera, Matessa, Freed and Remington (2002). Some issues have also been raised as to the accuracy of its time predictions (Olson and Olson 1990, John and Kieras 1996a), particularly if the original time values used in the model are not representative (Kieras, Wood and Meyer 1997).

Natural GOMS Language (NGOMSL) and GOMSL
Natural GOMS language, or NGOMSL, is designed to make GOMS easier to read. Kieras defines NGOMSL as “a structured natural-language notation for representing GOMS models and a procedure for constructing them” (Kieras 1996a). The model is based on the Cognitive Complexity Theory or CCT. The CCT is based on the suggestion that working memory triggers the execution of rules and operators at a constant rate, which also changes the contents of working memory (Kieras, 1996b).
NGOMSL breaks complicated conditional statements (decision making) into several steps based on the assumption that humans use several cognitive steps to process these conditions. Operators are classified as internal (mental) or external (perceptual motor), and only mental processes relevant to a given task are modelled.

While NGOMSL is easy to read, because it is based on the CCT it is also claimed NGOMSL can predict not only execution time, but learning time as well. Formulas for calculating learning time can be found in Kieras (1997).

A more recent computational form of NGOMSL called GOMSL is proposed by Kieras (1999, 2004a) for use with his GLEAN3 tool. Though almost identical in their description of tasks, GOMSL does offer additional syntax for how data is described, and make slight modifications to the memory operators. A full description of the differences, and the rationale for using NGOMSL instead can be found in 3.3.7.

3.3.5 Capabilities of GOMS

Over the years critics have argued that there are several limitations with the GOMS method. This has prompted the extension of GOMS with new variants and extensions. Based on issues raised by Olson and Olson (1990), Karat (1988) and others, this section discusses some of the issues raised in relation to GOMS’s capabilities.

Structuring documentation

Since GOMS documents the procedures necessary to complete tasks successfully, some researchers have suggested using this information to describe the tasks to users. Studies by Gong and Elkerton, (1990), Elkerton (1988) and Kieras (1988) suggest GOMS based manuals are less taxing and more efficient.

Ensuring Consistency

Kieras (1997) measures consistency in NGOMSL by comparing the number of similar statements between different methods.
Kieras (1997) used his NGOMSL method defined consistency, as a feature evident in a GOMS model as follows:

"Consistency can be measured in terms of how many statements have to be modified in order to turn one method into another, related, method." (Kieras 1997)

Kieras (1988) also provides a process for determining consistency between different processes. This process involves comparing the number of GOMS statements that are identical between two processes, described as 'method' consistency. Carroll also suggests that task decomposition methods like GOMS are able to measure low-level consistency but are inappropriate for contextual issues. It is clear then that GOMS is able to evaluate at least some of the consistency related to a design.

**Representing non-skilled or casual users**

It is GOMS's reliance on 'skilled' behaviour that is frequently criticised (e.g. Lewis and Wharton, 1997). For instance, Ivory and Hearst (2001: 496) argue that GOMS's predictions are "limited to error-free expert performance." There are several instances of this in GOMS studies. For instance John (1990) assumes that the users are experts in a telephone operator study. Gray, John and Atwood (1993) argue that the prediction of expert behaviour is a strength of GOMS, especially since their study concentrated on expert behaviour.

Bovair, Kieras and Polson use a production rule implementation of a GOMS model to address this issue. In order to characterise different levels of behaviour they define novice behaviour as follows:

"A novice user has acquired all the methods and can execute them correctly, but has not had a chance to extensively practice them" Bovair, Kieras and Polson (1990: 14).

They assumed that the expert user would need less prompts and a more compact set of rules for conducting a task. Their definitions differ significantly from other definitions of novices and experts. For instance, Vaubel and Gettys provide the following definitions:
• Novice - “an individual who has never used a word-processor of any kind”
• Intermediate – “individuals who have some word-processing experience”
• Expert – “an individual who has spent a considerable amount of time doing word processing and who is extremely efficient due to his or her extensive knowledge of commands.” (Vaubel and Gettys, 1990: 104)

Bovair et al’s definition of novice is as a far more experienced user than Vaubel and Getty’s definition. Interestingly Vincente (1999) suggests that using expert behaviour as the basis for task analysis has a number of benefits. He states:

“By adopting such an approach we should be able to design computer-based information systems that deliberately induce and support expert action, thereby leading to gains in cognitive efficiency.” Vincente (1999: 186).

Whether GOMS task-analysis is better off by assuming ‘expert’ behaviour is still in contention.

**Difficulties in the learning and usage of GOMS**

John and Kietas (1996b: 304) defend GOMS against claims it is expensive and unwieldy to implement. They argue that many GOMS studies have used GOMS at a research level and that it is quicker when used as an established technique. They conclude that KLM and CMN-GOMS are able to be taught in a single university level lesson. NGOMSL takes longer to learn and use, owing to its precision and focus on working memory. CPM-GOMS takes time to coordinate the perpetual, cognitive and motor-processing activities it uses. Despite this Lewis and Wharton suggest that though other methods might be easier to conduct, such as a cognitive walkthrough, it may be worth investing extra time learning the GOMS framework.

**Analysing learners and predicting learning time**

Another significant criticism is that GOMS is unable to predict or model the behaviour of learners. For instance, Rosson and Carroll (2002) argue that by focusing on the individual operators that make up tasks, GOMS models ignore significant factors of human behaviour such as problem solving and learning. Guzdial (1999) suggests that, because learners have ‘unknown’ goals, GOMS
models are an inappropriate way to model learner's behaviour. In another instance Karat (1988: 901) argues that the original GOMS model has "little to say about learning" or how what is learnt is reapplied on another computer system. Nielsen (1993) also argues that GOMS cannot determine the learnability of an application (how long it will take to learn). However many of these criticisms have been addressed by Kieras's (1997) NGOMSL model, which provides formulas for the prediction of learning time.

Using pre-determined time values corresponding to recurring NGOMSL statements across several tasks, Gong and Kieras (1994) were able to predict learning time with reasonable accuracy. However John and Kieras (1996b) argue that these predictions are limited when base-line times (the time needed to execute these tasks) are not available or when no training is needed. They also argue that these methods ignore any domain knowledge needed to perform a task. Since no baseline times for the tasks in the studies were available or predetermined for this study, no learning time is calculated.

**Identifying and handling errors**

GOMS has been criticised on several occasions for only addressing 'error-free' behaviour (Lewis and Wharton, 1997, Karat 1988). Since users spend up to 25% of their time making and recovering from errors (Carroll and Olson 1988) and even expert users have error rates as high as 20% (Kitajima and Polson 1995), this is an important performance consideration. But since GOMS analysis occurs without user testing Young and Whittington (1990) argue that GOMS is precluded from predicting errors satisfactorily. This is not a criticism that should be levelled at GOMS alone. Johnson (1999) argues that very little has been done to provide methodological support for error analysis.

However Card Moran and Newell (1983) who invented GOMS argued that since there are many different stages a user must progress through, errors are in fact identifiable. Others such as John and Kieras (1996a) identify three possible ways GOMS can address errors. These are:

- "preventing users from making errors"
• "predicting or anticipating when and what errors are likely to occur given a system design"
• "helping the user recover from errors once they have occurred"
  (John and Kieras, 1996a: 300–301).

However, Lerch, Mantei and Olson's (1989) study of skilled performance demonstrated that GOMS could be used to anticipate some types of error. However more recent work by Wood (2000) has demonstrated that GOMS can be used quite successfully to anticipate and prevent errors in design. Wood proposed a series of patterns and heuristics for classifying and identifying several types of error in a GOMSL model. These covered many types of error such as when too much information must be remembered, or when the user must execute complex calculations. Wood tested out these procedures on an e-commerce site, by building what he describes as a "pathological" design built to promote error prone behaviour and comparing it to another design designed using GOMS to avoid errors. By doing so Wood was able to demonstrate that GOMS is able to greatly reduce the error rate (by as much as 91% in some instances) when applied to the design process.

Other ways suggested by Wood (2000) include setting up exception mechanisms, like those used to recover from system errors, into GOMS models, and to treat the procedures for error recovery just like procedures for other goals. Treating error recovery as just another goal was proposed some time earlier in John and Kieras (1996a). Wood's work demonstrates that while this remains an ongoing issue, the prediction of errors is not an exclusive feature of user testing alone.

**Modelling cognitive processes**

Though cognitive modelling has been around for some time, there is still room for substantial improvement. Optimistically, Ritter and Young (2001) suggest that cognitive models might eventually be so detailed that they can actually replace the user in user testing. This is perhaps 'science-fiction' but it highlights the ambitions of cognitive analysts. For now at least, the human mind and all its functions are too complex to predict every aspect of behaviour accurately. GOMS may be an approximation of human behaviour, but is it a good one?
This issue is further complicated by those who believe that users are unaware of their own mental operations. For instance, Kieras (1997) states that the history of cognitive psychology suggests that “people have only a very limited awareness of their own goals, strategies and mental processes in general.” (Kieras, 1997: 744).

But even if this is true in an absolute sense, this does not undermine the value of cognitive modelling or GOMS. People do not need to be aware of their own decision-making processes for analysts to usefully anticipate aspects of their behaviour. Wood (2000) demonstrated how identifying error-producing behaviour in GOMS representations could significantly reduce error rates in an e-commerce application. Successful applications of GOMS and its variants to predicting execution time and other factors, repeatedly demonstrate that model based evaluation is as valuable a tool in designing for user behaviour, as testing with actual users.

**Modelling parallel processes**

CPM-GOMS uses parallel operators to represent physical mental, and cognitive processes that may occur simultaneously. This only applies to a single task. What about multiple tasks occurring concurrently? Salvucci (2001) provides an example where a user drives a car and uses a phone at the same time, since each task influences the other. Salvucci believes that GOMS does not have the power to represent concurrent tasks. However Kieras (2004a) argues that GOMS is capable of representing asynchronous tasks. This is demonstrated in a study by Kieras and Santoro (2004) where warship personnel must conduct a number of tasks and interact with other personnel at the same time. Kieras (2004a) also argues that asynchronous events in GOMS can be accommodated via IF-THEN statements dependant on the psychological state of the user. However he acknowledges that this is an area that does require significant further research.

**Individual differences**

Cognitive modelling generally focuses on the mental operations of a generic user, who is typically considered an expert. In the original GOMS text, Card et al (1983: 119) concluded that the design of the text-editor in relation to speed, was not “comparable to, and not dominated by, the effects of individual differences.”
Cognitive modelling is often criticised for ignoring the contextual factors associated with design. Olson and Olson (1990) stress that cognitive approaches like GOMS are unable to deal with other factors that influence behaviour such as user’s fatigue, the individual differences between the users, and the mental workload to perform a task. Stenning and Gurr (1997: 126) argue that ‘cognitive science’ is embarrassed by the fact that ‘People do differ in how they go about reasoning’.

Those arguments aside, GOMS is extremely useful in the early stages of design. Olson and Olson acknowledge that cognitive modelling is:

"useful in both initial design (it can narrow the design space and provide early analyses of design alternatives, evaluation and training". (Olson and Olson, 1990:222)

Therefore GOMS is useful in early stages of design, including task-analysis and comparing different designs. As for the ‘mental workload’ issue, this was addressed by the NGOMSL model documented earlier in the chapter.

Contextual factors

The importance of contextual factors is highlighted by Karat who notes:

"The usability of a product is not an attribute of the product alone, it is an attribute of interaction with a product in a context of use."

Karat (1997: 692)

Rosson and Carroll (2002) also argue that techniques like GOMS do not address all the issues that influence human behaviour, since they miss out important factors such as “the structure of work and organizations, and the experiences of learning and problem-solving.” (Rosson and Carroll, 2002: 236). Since contextual factors are so broad, it is left to the requirements analysis to provide the constraints on the design of the system. Kieras et al (1995: 93) defend the GOMS approach. They argue that GOMS can predict the ‘procedural aspects’ of usability, namely Simplicity, consistency, and efficiency of the design, and thus has considerable “value in guiding interface design.”
As in the section on Individual Differences, GOMS remains a valuable technique for the early stages of design. Contextual factors are important, but are best covered by a comprehensive requirement analysis such as the Scenario Based Design and Cognitive Work Analysis techniques documented earlier in this chapter.

**Representing visual layout**

One obvious limitation of the GOMS technique is the fact that it does not directly address the visual aspects of design. For instance, GOMS will not address issues of usability such as the 'legibility' of typefaces (Kieras et al. 1995). It can however tell you how to group icons, structure a menu and which buttons to put on a toolbar (Dumas and Reding, 1999). This is because these structural issues may be determined without looking at the screen. Chua (1994) for instance, was able to evaluate different screen layouts using NGOMSL. They found that the error rate was only 8%. It is fair to say that while GOMS models can address user interface structure, the visual and aesthetic aspects of graphical display, are not addressed by GOMS. These issues are better designed by using style guides.

**GOMS and web design**

It has been lamented that few tools exist to enable the analysis of human performance and usability for web sites (Chi Pirolli, Pitkow, 2000). Ironically, the creators of GOMS Card et al (2000) believe GOMS (and other cognitive engineering models) are ill suited for web design and analysis due to an inability to adequately represent unstructured user goals, multiple means of navigation and the influence of content on the users' behaviour. Ivory and Hearst (2001) also raise the point that GOMS is incapable of representing the different user types who use the web.

Even so, Card et al (2000) conclude that Web-Systems can be better built using GOMS to address design issues such as response time. Byre et al (1999) also suggests that task-analysis techniques (like GOMS) should be used to analyse web sites at a higher level of abstraction concentrating on more on the tasks and less on the behaviour.
GOMS tools

According to Van Setten et al (1997), there is no point in using a tool if the underlying theory is not sound. Despite arguments against several aspects of GOMS, many researchers have attempted to build a GOMS based design tool. Baumeister et al (2000) argue that as of that time, none of the methods including GLEAN and QGOMS could be considered to be an ideal GOMS tool.

GOMS tools include Quick GOMS (QGOMS), a way of building and analysing GOMS models rapidly (Beard et al 1997). QGOMS is not as accurate (Beard et al 1996), nor as thorough (Baumeister, et al 2000) as other GOMS tools. A different tool called CRITIQUE (Hudson et al 1999) allows designers to automate the construction of KLM (keystroke level models). It is recommended for use with the cognitive walkthrough technique (Hudson et al 1999). Another tool called GLEAN (GOMS Language, Evaluation and Analysis (Kieras 1999) supports GOMSL modelling (the computational form of NGOMSL). GLEAN has been criticised as being too time consuming when building interface simulations (St Amant and Reidl, 2001), and limited when it comes to addressing learning and errors (Ritter and Young, 2002). There are older tools, such as CATHCI and TYPIST. The most recent tool is John’s (2002) APEX tool, which is based on her CPM-GOMS techniques.

One limitation of GOMS tools is that they do not provide the necessary support for evaluating usability (Baumeister et al 2000). A second limitation with GOMS tools is that the behaviour of the system is not extensively represented. For instance, John (2002) decided a single system component alone is not enough to represent the system in a GOMS model when building the APEX tool. However, APEX only provides a limited representation of system behaviour.

According to Bhavani and John (1998) if a tool is unsuitable for a task the user can either change the process or move on to a new tool. Diaper (2004) argues that while it is desirable for task analysis input to be in the form of one or more software engineering representations, none of the task analysis methods described in their text do so, including HTA and GOMS. Neither do any of the tools provide
this kind of representation. It is for this reason that the approach proposed in this
study is not reliant on any of the currently available GOMS tools.

3.3.6 Field studies and validation of GOMS

One advantage of GOMS is that is extensively documented and tested. Raskin
argues that:

"In a field in which religious wars are waged over interface designs and
in which gurus often have widely varying opinions, it is advantageous to
have in your armamentarium quantitative experimentally validated, and
theoretically sound, techniques." Raskin (2000:73)

GOMS has been demonstrated time and time again, to be a valuable means of
analysing and predicting user behaviour. In text editing studies, Bovair et al
(1990) and Card et al (1983) were able to make accurate time predictions within
20% of actual subjects. Baskin and John (1989) accurately modelled expert
behaviour (using CPM-GOMS and KLM) for drawing shapes on screen. Lerch
Mantei and Olson (1989) were able to adequately predict some types of errors, the
'mental load' required to do the task, and execution time for financial modelling
systems. Bhavani and John used GOMS to successfully analyse the efficiency of
CAD (Computer Aided Design) tools. Chuah (1994) used GOMS to successfully
analyse the visual layouts of different systems finding an 8% difference between
predicted times and actual times.

There are as many, if not more, studies that demonstrate the accuracy of GOMS in
contexts other than software. These include studies on CAT-scans for radiologists
(Beard et al 1996), automatic teller machines (John et al 2002) and telephone
operators (Gray et al, 1992). This demonstrates that GOMS is a task analysis
technique and not just a software interface technique.

John (2002) applied CPM-GOMS to automatic teller machines and found an
acceptable 13% error rate in predicting user behaviour. A much larger study was
based on the tasks that TAO (Single Toll and Assistant operators) conducted. By
building an alternative prototype Gray et al (1992) found that the proposed system
(the prototype) was actually 3% slower then the original system. They were able
to prove this finding by examining the designs in greater detail. They obtained their times by actually observing users using their systems.

These studies highlight the broad application of GOMS. When using GOMS as a means of predicting execution times, a successful result will depend on the accuracy of the values used. Execution times can be found in Card et al (1983), John and Newell (1990) and Olson and Olson (1990).

Which GOMS model?

This study has already discussed why GOMS is an appropriate cognitive modelling technique during the early stages of design. But one important question remains. Which of the four GOMS variant techniques (KLM, NGOMSL, CMN-GOMS, and CPM-GOMS) should be used? Table 3-2 demonstrates the view of John and Kieras (1996a) on the scope of the various GOMS methods.

<table>
<thead>
<tr>
<th>Task Type</th>
<th>Design Information</th>
<th>Goal-Directed Routine Cognitive Skill With Passive or Active Systems</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functionality Coverage</td>
<td>Any GOMS</td>
<td>Any GOMS</td>
</tr>
<tr>
<td>Functionality Consistency</td>
<td>NGOMSL</td>
<td></td>
</tr>
<tr>
<td>Operator Sequence</td>
<td>CMN-GOMS NGOMSL</td>
<td>CPM-GOMS (see text)</td>
</tr>
<tr>
<td>Execution Time</td>
<td>KLM CMN-GOMS NGOMSL</td>
<td>CPM-GOMS</td>
</tr>
<tr>
<td>Procedure Learning Time</td>
<td>NGOMSL</td>
<td></td>
</tr>
<tr>
<td>Error Recovery</td>
<td>Any GOMS</td>
<td>Any GOMS</td>
</tr>
</tbody>
</table>

Table 3-2 Classifying GOMS techniques John and Kieras (1996a: 294)

3.3.7 Natural Language GOMS in detail

As mentioned earlier NGOMSL was designed by Kieras (1997) to be human readable. This is evident by the ways in which the GOMS components are represented. The notation for these is as follows:

Goals:

Sub-goals in NGOMSL are initiated from a parent goal as follows:

“Accomplish Goal: <goal description>”
Methods:
The methods themselves represent a sequence of actions to complete a given goal. In NGOMSL they use the following syntax to represent a method:

Method for Goal <Goal Description>

This statement is then followed by a set of sub goals and or operators.
The minimum number of statements in a NGOMSL method is three.
- An ACCOMPLISH GOAL statement first calls the method.
  In GOMSL this can be abbreviated to AG (Kieras 2004a: 97)
- The method begins with a METHOD statement at the top of the method.
- A RETURN WITH GOAL ACCOMPLISHED statement ends the method.
  In GOMSL Kieras (2004a: 97) abbreviates this to RGD.

Selection rules
Selection rules are a set of mutually exclusive conditions for choosing from alternate methods to complete a goal. According to Kieras, (1997: 741). The basic format is as follows:

Selection rules set for goal
If condition then accomplish goal 1 <specific goal description>
If condition then accomplish goal 2 <specific goal description>

Return with goal accomplished

Operators
According to Kieras (1997) There are several kinds of operators in NGOMSL.
These are:
- External operators -- observable actions
- Mental operators -- internal actions (inferred by the analyst)
- Primitive or High level operators -- High level operators are actions that are broken down into lower level operators. Primitives are actions which are not broken down any further.
- Analyst defined operators. -- The analyst may choose to define his or her own operators. These are useful for complicated mental operators, which the analyst wishes to include in the model, but not define in further detail.
Unlike the KLM model, which has only one mental operator, NGOMSL has multiple mental operators. Some of these apply to Working Memory (WM) and Long Term Memory (LTM). These statements include:

- **Retrieve** – Fetch information from WM
- **Recall** – Drop unnecessary information, which is no longer needed, from WM.
- **Retain** – Store information in WM

Other mental operators include flow of control statements like Decide, and IF and Else. These are applied in the NGOMSL models in Chapters Five and Six.

**Task Descriptions and Task Instances**

In order to use the model as a task analysis technique Kieras distinguishes Task Descriptions from Task instances. Task Instances descriptions are a generic description of the task, and not an actual example of a task. The difference between the two is described in Table 3-3.

<table>
<thead>
<tr>
<th>Task Description</th>
<th>Task Instance</th>
</tr>
</thead>
<tbody>
<tr>
<td>The goal is to delete a piece of arbitrary text</td>
<td>The goal is to delete a piece of arbitrary text.</td>
</tr>
<tr>
<td>The starting location of text</td>
<td>The starting location of the text is line 10, column 1</td>
</tr>
<tr>
<td>The ending location of text</td>
<td>The ending location of the text is line 11, column 17</td>
</tr>
<tr>
<td>A find string for locating the beginning of the text</td>
<td>A find string for locating the beginning of the text is “Now is the”</td>
</tr>
</tbody>
</table>

**Table 3-3: Task Instances and Task Descriptions from Kieras (1997:742).**

**Counting NGOMSL Statements**

There are three statements in the NGOMSL model that count as single NGOMSL statements. These are:

RETURN WITH GOAL ACCOMPLISHED

STEP

IF, THEN (in a selection rule)

Due to the multiple statements in the DECIDE statement, it counts as one statement for every THEN or ELSE statement included.
3.3.8 The NGOMSL process

There are four steps involved in building and applying a NGOMSL Model. These are provided in the next table.

<table>
<thead>
<tr>
<th>The NGOMSL Process</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>STEP A:</strong></td>
</tr>
<tr>
<td><strong>STEP B:</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>STEP C:</strong></td>
</tr>
<tr>
<td><strong>STEP D:</strong></td>
</tr>
</tbody>
</table>

Table 3-4: The NGOMSL Process – (adapted from Kieras 1997).

In order to understand the method, the five steps in the NGOMSL process as described by Kieras (1997) are analysed in more detail.

**STEP A: Choose the Top Level Goals**

According to Kieras the top-level goals should be as high as possible, and should be able to fit into a unit-task structure (can be broken down into smaller tasks). Each of these ‘unit tasks’ should be able to be broken down into selection rules, which send control to the appropriate method. Kieras recommends that this unit-task structure is an extremely common one and would make a good template.

**STEP B1: Draft a method to accomplish each goal**

Depending on the level of analysis, Kieras suggests keeping the steps (goals and operators) as high as possible. He suggests using the KLM model as a means of determining operators from goals:

- Ignore complicated psychological processes
- Define high level operators (making assumptions is needed)
- Deal with underlying methods before determining selection rules
- Ignore ‘short-cut’ methods till later.
STEP B2: After completing the draft, check and rewrite as needed for consistency and conformance to guidelines

Kieras provides several tips for completing this step.

- Check on the level of detail and length of each method.
- Check that you have made consistent assumptions about user's experience with regard to the number of operators in a step.
- Identify the high-level operators you used; check that each high-level operator corresponds to a natural goal; redefine the operator or rewrite the method if not so.
- Maintain a list of operators in the analysis showing which methods each operator appears in.
- Check for consistency of terminology and usage with already designed operators.” (Kieras, 1997: 750)

STEP B3: Go to a lower level of analysis if necessary

The NGOMSL model allows you to analyse a task in detail if that proves necessary. The lowest types of operators such as the ones in the KLM model (e.g. keystrokes) should only be examined after the fourth level of detail. The analyst should keep on descending through the model until all the operators are primitives, or choose not to break down any high-level operators any further. In order to break down a high level primitive you convert it to a goal.

He also provides a set of recommendations for each step in the method. According to Kieras (1997) each step should have:

- Only one accomplish goal operator
- Only one high level operator
- One external primitive operator
- No more than one primitive operator for ordinary and novice users
- A sequence of primitive operators, but only if an expert user can perform these actions without decisions or sub-goals.

Mental operators should be used as follows.

- A LOCATE operator before pointing with mouse
• A VERIFY operator when the user needs to make sense of feedback
• A THINK OF or GET_FROM_TASK before a user obtains task-parameters information.

STEP C: Document and check the analysis
In order to document the analysis you should provide:
• Lists of external operators used
• Descriptions of analyst defined operators
• Assumptions and judgement calls
• Contents of task descriptions for each type of task.

In order to verify the model you may either follow the analysis by hand, or if possible automate the process in software.

STEP D: Check what impact the judgement calls and assumptions
It may be worth building several models to see how these judgements affect the design. It will not be necessary if two design models are based on the same assumptions and calls.

3.3.9 The differences between GOMSL and NGOMSL
This study uses NGOMSL as the basis for its natural language requirements in two models documented in later chapters. However Kieras (2004a) has updated the NGOMSL notation to make it more computationally executable. This new format is called GOMSL.

The core changes to GOMSL (from its predecessor) are as follows:

• GOMSL has an enhanced object-property representation.
• New rules for intra-step and inter-step operators, when to move on to a new line.
• Renamed memory operators
  o Forget is now called Delete
  o Retain is now called Store
  o Retrieve from LTM is now Recall from LTM.
• Support for abbreviations – Method for Goal can be MFG.
• Some syntactical changes including the use of underscores instead of spaces in command names.
• Pseudo-arguments and pseudo parameters – similar to variables but without the “scoping rules” of actual programming languages. (Kieras 2004a: 101).

This study used NGOMSL as the basis for models in later chapters over GOMSL for the following reasons.

• The VCR model used in Chapter 5 is constructed in NGOMSL
• The object property representation was not necessary to convey the tasks carried out in the studies, since NGOMSL was used as the basis for a GSE conversion.
• The abbreviations are easily accommodated in NGOMSL, just by changing the text.
• Changes to the memory operators, had little impact on the NGOMSL models used in this study.
• Since NGOMSL was being used as the basis for the natural language requirements for a GSE conversion, it was not necessary to use the more computationally succinct syntax of GOMSL when building the GSE models.

So while acknowledging GOMSL and its concise computational format could just as adequately be used as a basis for a GSE conversion, it was not necessary to do so.

GOMS and Guidelines

One of the interesting aspects of the NGOMSL model is in step B2. Kieras (1997) suggests using guidelines in conjunction with the GOMS method. This highlights the difference between task-analysis models and guidelines and heuristics. Raskin (1997) suggests that eventually the GOMS tools will eventually be good enough to replace guidelines and heuristics altogether. Raskin describes this as a move from heuristics to cognetics, since GOMS is a cognitive model. Whether an automated GOMS process such as APEX (John et al 2002) or GLEAN3 (Kieras 1999) becomes capable of building perfect interfaces remains to be seen.
3.4 Software engineering models

According to Lewis and Rieman (1994), there is no point meeting specifications, if what is being built is a failure. This implies that meeting functional requirements alone will not ensure a ‘good’ or successful product. It is increasingly clear that the design of a system is dependant not only on its functional requirements, but usability issues as well. However, Ormerod and Shepherd (2004:348) argue that developers must also “preserve control over specifying the functionality of their systems” but not at the expense of user-centred approaches. Now that user principles, cognitive modelling and task analysis have been discussed it is important to look at software engineering, and why methods aimed at addressing the user’s needs are not necessarily well represented when designing the system.

One reason why software engineering approaches do not always address usability adequately is that the overall process itself poses problems. This is particularly true of the traditional approach to software engineering called the ‘waterfall approach’. Rosson and Carroll describe this process as follows:

“Software engineering is organized into a series of modular phases, beginning with the analysis of the functional requirements and continuing through software design, implementation, testing and maintenance. Each phase produces one or more documents that are handed off as a specification for the work of the next phase.”

(Rosson and Carroll, 2002: 6)

The waterfall approach is basically a series of steps beginning with a requirements analysis (Lewis and Rieman 1994). These requirements are transformed into system specifications and later implemented as a set of components according to these specifications. However, the linear nature of this approach poses problems in terms of cost and redesign, especially for user interface designers (Pew 2000).

The major problem is that even though it is possible to return to earlier stages of the process to make changes, the costs involved in doing so can be exorbitant especially since these costs increase significantly the further back in the process
the developer went (Diaper 2004:24). Thus the weakness of the waterfall process is its inability to “cope with change” (Sommerville 2004:102-7). According to Pew (2003) Interface designers found the waterfall method limiting, since it did not support the building of prototypes of the final design. This point is reinforced by Sommerville, who states:

"More and more systems are interactive systems with complex graphical user interfaces. Experience has shown that the waterfall model is particularly inappropriate for this type of system development." (Sommerville, 2004: 102-6).

However, the waterfall model is not the only software development process available. Another significant software process is the evolutionary development process that is “characterised by the early use of prototypes.” (Schmietendorf, Dimitrov and Dumke 2002). Essentially prototypes are built early, and modified over time until the system functionality matches the user’s needs. While this is more likely to produce software that meets the user’s requirements, other functional requirements may be neglected and the constant change may later lead to maintenance problems (Sommerville 2004).

Another process model is the spiral model proposed by Bohem (1988). Unlike the waterfall model the spiral model accommodates prototyping, evolutionary design and development cycles (Guimarães and Vilela 2005, Raccoon 1997). The spiral model is a four-stage process: objectives are determined, risks are analysed, development activities take place, and then plans are made for the next iteration. The key feature of the spiral model is the analysis of risks for a given phase of development. When issues are identified they can be resolved through resolution activities such as prototyping (Sommerville 2004).

Modern software engineering processes include the Rational Unified Process (Kroll and Krutchten 2003), the Open Process Framework (Firesmith and Sellers 2002), and Catalysis (D’Sousza and Wills 1998). These provide guidance on design activities at various stages of the software life cycle.
So how are usability problems better addressed in new incremental approaches as opposed to the traditional waterfall model? There are several major advantages. In methods like the Rational Unified Process, the development occurs in iterations, and development cycles (Kroll and Krutch 2003), meaning the product is refined in successive passes. Thus the development team can build various types of prototype, test these prototypes with actual users to identify problems, and use their findings to refine their design in the next iteration. Prototyping the user interface is particularly important as it can be used to “pose questions and evaluate options from the end-user viewpoint” (Firesmith and Henderson-Sellers 2002:201). In the Open Process Framework, Firesmith and Henderson-Sellers (2002) assert that the main criterion for evaluating the user interface is Usability.

The key difference with the waterfall model and modern methods is the repeated testing and refinement. Testing may also include heuristic testing, walkthroughs, and any of the other methods identified in Chapters 1 and 2, although they all have pros and cons including time, expense, and access to experienced personnel. However the common aim is to eliminate usability problems over successive iterations.

Despite the evolution of software processes there are other factors that ensure usability issues are not adequately addressed in software engineering. Diaper (2004) has noted that task analysis techniques do not produce an adequate functional representation of the system. Kieras (2004b:46-2) argues that system requirements are often thrown ‘over the fence’ to the developers, who then throw their design ‘over the fence’ to usability experts who are then expected to ‘put a good interface on it.’ However it is important to state to imply that software developers clearly do not choose to ignore ‘usability’ issues. The move towards more iterative processes and the recognition of usability as a design issue clearly indicate the opposite. However there are reasons why HCI issues are not always addressed adequately.

John (2004) for instance, provides a number of reasons for why HCI is not necessarily addressed in software engineering. John states that:
“except where government regulation requires, the HCI aspects of software engineering are optional and often fall victim to the pressures of deadlines and scarce resources. Where government regulation requires, the HCI aspects of software engineering are still extremely dependent on the informal processes and the particular training and skill of individual HCI professionals. Some phases of development have no methods for HCI input and few tools exist to help regularize the contribution of methods we do have.” John (2004: 285).

It is well established that the cost of re-engineering software is high (Karat 1997), but understanding the user's tasks is paramount. However developers are limited by the time, costs and available expertise. This is why Perlman (1996) suggests that projects with limited skills and resources need to concentrate on practical methods that can be applied. The question to ask then is: Can GOMS be practically integrated into a software engineering technique such that the user's tasks and usability issues are addressed?

This section discusses how a system engineering model might better represent the total set of system requirements (user and system). Usability is so strongly rooted in the interaction between user and machine that to not adequately provide a 'total' representation of the system limits the capacity to address it. Thus, according to Kieras and Polson (1985: 379), to fully characterise the interaction between the user and a device requires an “explicit and formal representation of the behaviour of the device.”

Ultimately, any of the software development processes mentioned are reliant on some form of representation methodology in order to capture the functional requirements and transform these into a design. Though not all processes are dependant or specify a particular modelling technique, others are reliant on an established set of notations. But to what degree can system-engineering models adequately represent the user's involvement? The next two sections discuss two alternate ways of modelling the functional requirements for a system, UML and GSE, and their suitability for modelling the total set of functional requirements (including the user's task requirements) for a device.
3.4.1 UML, The Rational Unified Process and Related Methods

The Unified Modelling Language or UML is not a software engineering process in its own right. Instead UML is 

"a general-purpose visual modelling language that is used to specify, visualize, construct and document the artifacts of a software system."

Rumbaugh, Jacobson and Booch (1999).

UML is essentially notation for software engineering, which may be employed by software engineers using methods that employ UML notation. Scogings and Phillips clarify this distinction between notation and method by defining UML as 

"a collection of semi-formal graphical notations which can be used to support the design of software packages. No particular methodology is specified." Scogings and Phillips (2001:70).

UML provides software engineers with a number of different diagram types. Use Case diagrams represent the interaction requirements between those involved with the system and the system itself; Activity diagrams describe the business processes; Sequence diagrams show the flow of activity and Class diagrams describe the structure of the system (Lunn 2003).

UML is a notation and not a process in its own right. However RUP, or the Rational Unified Process, one of the most commonly applied software development processes, (Zuser, Heil and Grechnenig 2005) is integrally linked to UML notation. In fact Krutch (2000) describes RUP as a "guide to the effective use of the UML for modelling" because it describes when and which models are needed, and how to build them. RUP can be classed as an evolutionary software development process since the process is reliant on iterative and incremental development (Schmietendorf, Dimitrov, Reiner and Dumke 2002). The procedures in RUP are described as "use case driven" meaning they rely heavily on the specification and application of use cases as the basis for development. They are also risk driven, as the process aims to identify and address any risks that might appear at a given stage of development. While a detailed examination
of RUP is not pertinent to this study, it does demonstrate how UML integrates into a commonly applied software engineering process.

In fact, UML is now so widely applied, that it is easy to see it as the 'only’ viable engineering notation for software engineering (Bell 2004). Bell warns against a number of unreasonable expectations associated with the application of UML by engineers, particularly those that see UML as a cure-all. Bell admits the same could probably be true of other software approaches except that “no other technology has so quickly and deeply permeated the software-engineering lifecycle quite like UML” Bell (2004: 74).

According to Medvidovic, Rosenblum, Redmiles and Robbins (2002), UML’s primary strength is that it is an excellent means of modelling object-orientated concepts, when all the major system elements can be represented as objects (including interaction between them). UML’s second major strength is its flexible design philosophy (Medvidovic et al 2002) in the sense that the notation itself can be applied by different designers and developers in whatever manner they deem fit. Of course this will be constrained by whatever other frameworks and methodologies they apply.

However UML is not without potential limitations. Firesmith and Henderson-Sellers (2002:126) argue that the current UML meta-model is “somewhat weak on semantics and still, largely, as describing syntax.” UML can also be difficult to read and interpret (Arnowitz et al: 2000). Eichelberger (2003) for instance, argues that UML does not say anything about how to produce readable diagrams and that many of the 43 related UML tools they studied generated horrible layouts. Medvidovic et al (2004) argue that the architect’s job may be made more difficult by details muddled by abstractions between different types of diagram.

However, since this study aims to address usability issues by using a total model of system behaviour, there are two serious obstacles to using UML. The first issue is how requirements are analysed and represented. In fact Kholker, Krisnam, Shiotri and Venkatesh (2005: 77) argue that it is the inadequate analysis of requirements in UML that is responsible for many defects in software
development. The problem relates to how UML uses use cases in order to represent requirements.

Use cases are typically employed at the task analysis stage. According to Biddle, Noble and Tempero (2001:7), the general idea behind a use case is to "represent intended sequences of interaction between a system (even if not yet implemented) and the world outside that system". These cases map the tasks, based on their relationship to 'actors' who can be thought of as participants in the system. Customers, checkout operators, managers, and external suppliers may all be examples of 'actors' in use cases for a supermarket system. Bustard et al (2000:98) describe actors as agents who "trigger the functions of the system" and use cases as descriptions of a system transaction.

Unfortunately UML is incapable of representing all the behaviours in functional requirements and designs (Dromey: 2002a). Bustard et al (2000) also concede UML is insufficient for determining requirements and needs an additional analysis stage.

These limitations are demonstrated in a case study of a remote medical care system analysed by Glinz (2000) who found:

- UML use case models cannot specify requirements where the system (and not the actors in the cases) initiates the interaction.
- UML could not represent the structural relationships between use cases in a straightforward or easy way.
- There was no appropriate way to represent interaction between the use cases.
- UML was unable to adequately express self-dependant system behaviour.

Thus system state could not be adequately represented.

UML is not a task analysis technique (see Diaper 2004). To elicit the relevant information for building use cases, Benyon and Macauley (2002), suggest using Rosson and Carroll's (2002) Scenario Based Design approach. However, UML is also criticised as being incapable of modelling the user's task requirements.

This section has raised issues related to the use of UML as means of building a total system representation incorporating the user and system's behaviour. However it is important to stress that UML is not specifically designed to be either formal or precise in its application. In a rather frank assessment, Rumbaugh et al provide an excellent statement to this effect:

"UML is messy, imprecise, complex and sprawling. That is both a fault and a virtue. Anything intended for such widespread usage is going to be messy." Rumbaugh, Jacobson and Booch (2005: 12).

It is clear that UML has proved its value in numerous projects and applications well beyond the scope of this study. It is also not possible to address the numerous extensions to the UML specification designed for a wide range of purposes. But this section has raised issues that question whether UML is suitable for integrating both the task and system requirements into a ‘total model’ that might be better used for addressing usability issues.

Thus this study proposes using an alternative. The next technique, the Genetic Software Engineering model or GSE, relies on a single type of meta-model, the behaviour tree for all aspects of its representation. It is hierarchical in nature (compatible with task-analysis approaches HTA and GOMS) and represents the user alongside the other functional components in the model.

3.4.2 The Genetic Software Engineering model

GSE stands for Genetic Software Engineering, and despite its name has no connection with human genetics. According to its creator Dromey (2002a), its purpose is to provide a simple and common graphical notation for representing requirements and designs.

There are several issues the GSE model seeks to address:
- A formal and useful means of getting from requirements to design
- The complexity and variety of current notations
• Requirements/design and incompatibility
• The high cost of software change.
• Identifying requirement problems early.

Thus GSE is intended to be formal, simple, compatible, cost-effective, and preventative, in relation to software design. It should be noted that though this thesis is focused on software design, GSE behaviour trees are not limited in their applicability to software systems, as will be demonstrated in the VCR and mobile phone studies in later chapters.

3.4.3 What is GSE?

GSE is a five-step process for transforming system requirements into design. However, it employs a single diagram type (the behaviour tree) as its form of representation.

Figure 3-3 demonstrates the structure of a GSE node. The first field in the right-most box corresponds to the name of the component. The second field is the behaviour of this component. The left-most box, the tag, is used to label and sort the nodes in a GSE model.

There are two major types of behaviour that determine the processes in a GSE model. The first of these is the realisation of a state by a component (node 1 in Figure 3-4). The second type is an event (node 2 in Figure 3-4) where the user ‘component’ is pushing a button.

Figure 3-4: Primary GSE Objects
Though GSE maintains a single type of component representation (see Appendix A), there are many other behaviours that can be recorded (e.g., adding a value) and types of components (e.g., the system component). Unlike a flow chart, the GSE process is hierarchical in nature. GSE does provide connections between components in a similar form. There are ‘AND’ and ‘OR’ conditions, and provision for loops in the system. An important aspect of GSE is demonstrated in Figure 3-4 where the user is represented as a component. This is a significant difference to other notations such as UML, since the same notation represents users and devices as components.

There are five steps to the GSE process. These are as follows:

1. Develop a statement of requirements
2. Translate each functional requirement produced in step 1 (including constraints), into its equivalent representation in one or more behaviour trees.
3. Integrate into the design, one at a time the behaviour tree representing each functional requirement.
4. Transform the design, one at a time, the behaviour tree representing each functional requirement that was produced in step 2.
5. Project from the design behaviour tree the behaviour trees of each individual component.
   (Dromey, 2002a)

This process is better demonstrated in the case studies, but it is important to note that behaviour tree notation is used at every stage of the process, and not just the initial statement of the requirements.

3.4.4 The Differences between GSE Behaviour Tree and UML Representations

UML is a collection of notations and diagram types for software engineering. GSE is a software engineering process that employs a behaviour tree representation. This study has previously discussed usability principles and task
analysis techniques and how they might be integrated into a total system representation. This section has introduced two core software engineering approaches; those based on UML notation and the more recently proposed GSE behaviour trees. Since the integration of the user and system requirements are paramount, some thought should be given to which of these approaches best serves this purpose.

While examining the formal representation of user complexity, Kieras and Polson (1985: 379) stated that to fully characterise interaction between a user and a device an “explicit and formal representation of the behaviour of the device is required.” They also specified six properties that such a representation would need to have:

1. “the representation should have well-defined formal properties so that its correctness may be explicitly determined”.
2. “the representation should make interactive systems easy to represent”
3. “the representation should be modular, so that as much or as little of the device can be represented as desired”
4. “Hierarchical control or structural relations should be easily represented”
5. “The representation should not be committed to any particular hardware or software implementation”
6. “It should be easy to represent features of the system that have psychological implications” (Kieras and Polson, 1985: 379).

In order to clarify why GSE behaviour trees better suit such a representation these six criteria are discussed in relation to both notations.

**Well Defined Formal Properties:**

When it comes to determining the correctness of a UML model, UML is a semi-formal notation meaning, that some formality and rigour is lost in favour of being more flexible (Medvidovic 2002). In the latest UML language specification, Booch, Jacobson and Rumbaugh 2005 acknowledge a new companion specification called OCL (the Object Constraint Language) for specifying constraints such as Boolean expressions and pre-conditions. However, using OCL requires significant and additional training and expertise (Briand et al. 2005) adding additional complexity to the notation.
GSE behaviour trees do not need additional constraints such as those specified in OCL, since state-changes, pre-conditions, Boolean conditions, are already formally specified in the notation (see Dromey 2002a and Appendix A). According to Wen and Dromey (2004: 2++) this allows a “a direct and clearly traceable relationship between what is expressed in the natural language representation and its formal specification.”

The correctness of a model can be tested against its requirements, but UML is limited in this regard. UML is designed to be flexible in its application, but this is at a cost, namely a lack of well-defined execution semantics (Jager et al 1991), and ‘loosely coupled’ models (Glinz 2000). Rumbaugh, Jacobson and Booch (2005: 11) argue that while UML is “not a precise specification in the manner of a formal language” few programming languages are precisely defined and formal languages can be inaccessible.

**Interactive systems easy to represent:**

Given that usability and HCI are such important design issues, it is important to adequately represent interaction between user and system. In GSE, the user and system (and subsystems) are components in the same behaviour trees. In UML, the user is principally represented as an entity in the use-case diagrams, while the functionality of the system is represented in the other diagram types. This brings with it a number of limitations, particularly in representing how the system-initiated events influence the user’s behaviour (Glinz 2000). Unlike UML, GSE can represent the full range of user interaction in both the requirements and the design (Dromey 2002a), in a single diagram type, meaning the behaviour is not divided across representations.

Dromey (2002a) justifies the inclusion of the user or users as a component as follows:

“We suggest that it is important to properly capture the user interactions in both the requirements and the design. This is most simply done within the present/component state framework by treating the user (or users) as a component that can be in different states” Dromey (2002a: 10)
Since UML is an older and more widely discussed notation there is more material on the readability of UML diagrams (e.g. Eichelberger 2003) than there is on GSE.

**The representation should be modular: so that as much or as little of the device can be represented as desired**

It is impractical to represent the entirety of a system's behaviour (especially a large system) across all levels of detail, in a single diagram. Both UML and GSE can break diagrams up according to need. In UML model content can be split into "general-purpose hierarchical organizational units" called packages (Rumbaugh et al 2005:14). GSE notation (see Dromey 2002a) also accommodates the organising behaviour trees into systems and subsystems, and collapsible levels of detail.

**Hierarchical control or structural relations should be easily represented:**

Structural relationships are inherent in both UML and GSE representations. However hierarchical control, of the kind represented in task analysis diagrams such as HTA is not able to be easily modelled in UML, since it cannot express formal constraints, in the same manner that a GSE behaviour tree can. Secondly use case diagrams (where the user's task requirements are specified) cannot be hierarchically decomposed in the same manner as GSE behaviour trees. This is demonstrated by Glinz (2000) who argues that "**UML can neither express structure between use cases nor a structural hierarchy of use cases in an easy and straightforward way.**"

Thus while GSE is inherently hierarchical due to its tree-like structure, not all UML diagram types are. While much of the system structure can be hierarchically decomposed in UML class diagrams, this cannot be done for the use case diagrams.

**The representation should not be committed to any particular hardware or software implementation:** Neither UML or GSE Behaviour trees are tied to any hardware or software implementation. According to Rumbaugh et al (2005:11) UML is "**is intended for all kinds of things from business modelling to graphical**"
programming." Rumbaugh et al 2005:11. For instance, a recent extension to UML called SysML (see Hause, Thorn and Moore 2005), extends UML’s application from software engineering to other systems.

However UML in a traditional sense (without extensions) is closely aligned with the Object Orientated software development paradigm. GSE does not support the traditional Object Orientated design approach provided by UML, although Wen and Dromey (2004) note some similarities between behaviour trees and UML’s activity and class diagrams. This allows GSE to formally model functional requirements in a single representation. However, UML can produce specialised diagram types that focus exclusively on specific aspects of traditional software engineering (such as the UML class diagram) and additional technical issues not addressed by GSE diagrams.

**It should be easy to represent features of the system that have psychological implications:** Neither UML or GSE behaviour trees are explicitly focused on the psychological implications of system design. This is not surprising given that they are software engineering notations and not models of human performance like ACT-R or GOMS. Neither high-level task analysis techniques like NGOMSL models grounded on cognitive principles, yield ‘ready’ system representations. However unlike UML, since GSE has an integrated formal notation it can express all the constraints (decision making, preconditions for events and the completion of goals) contained in the NGOMSL task model. GSE can also model changes in the user’s behaviour, due to changes in the system state (and vice versa) since it integrates the two into a common diagram type. While this study cannot claim that GSE can either represent cognitive models to the detail of ACT-R or EPIC, the user’s behaviour can be more completely expressed in a GSE behaviour tree than it can in UML. This does not count extensions or additional diagram types that might be added to UML, but since the division of behaviour between the use-case and other diagram types in UML already limits the description of user-system interaction (Glinz 2000) and UML is complex enough, adding additional notation can only make this more difficult.
Based on the six above criteria, GSE behaviour trees are a far more suitable representation for the kind of ‘total model’ of interaction that this study aims to use. These reasons can be best summed up as follows:

- GSE models user behaviour and component behaviour in the same representation. Thus system-initiated interaction is modelled alongside user-initiated events.
- GSE incorporates formal constraints that can be used to model decision-making, and other conditions. UML is a semi-formal notation.
- GSE shares a similar hierarchical task representation as that used in HTA and NGOMSL. This is an advantage given we are trying to build a total representation of the system including the user’s task requirements.

The aim of this section is not to demonise UML, which is of inestimable value to software and system designers. However GSE behaviour trees are better suited to integrating the user and system requirements in a single representation. Given that the usability of a system, including the attributes outlined in Chapter 2, is determined by interaction between both user and system, this study proposes that just such a model is better suited for identifying usability problems during the design phase.

3.5 Summary – the future

This chapter has introduced task modelling, GOMS and software engineering. The high level NGOMSL notation demonstrates that cognitive principles can be embodied in a high-level task model to predict aspects of the user’s behaviour.

The Software Engineering section argued that a total system model (incorporating user and system behaviour) could be constructed in a single representation. So can a GSE model based on an NGOMSL-structure be used to find usability problems based on those attributes outlined in the previous chapter? The next chapter proposes a methodology for how these problems can be found in a GSE model, and outlines a testing methodology for determining the success or failure of just such an approach.
CHAPTER 4
METHODOLOGY

"Usability is an important part of the validation process and usability issues are too important to be left out as their impact on the overall quality is considerable."

(Garg 2001: 6)

4.1 Introduction

This study is aimed at identifying usability problems before deployment, using a new technique based on quality producing attributes, cognitive and system design. Older techniques are typically split between those that need actual users, such as user testing, and those based on knowledge or pre-defined HCI recommendations, such as heuristic evaluation. Few however seek to consolidate cognitive principles (such as Card et al’s rationality principle) and system orientated design methodologies such as UML into a single approach. The NITRD (2003) however, have recently recommended that analysts seek to consolidate cognitive principles and system design into a pre-deployment ‘total system’ model that is capable of making better predictions on the overall usability of a system.

The primary objective of this study, therefore, is to identify and prevent usability problems prior to deployment using a new integrated approach to design and evaluation. This method must be tested in relation to its effectiveness as a usability evaluation technique. According to Ebling and John (2000: 289), effective usability evaluation “cannot simply answer with a ‘yes’ or ‘no’ but must provide detailed information about why the design does not work as anticipated, or at least what problems the users experience.” Put simply, this study must prove that the proposed technique is an effective/pre-deployment method for identifying and addressing usability problems.

Therefore this study aims to:

- Determine whether quality attributes can be used to predict and prevent usability problems.
• Establish if a cognitive modelling technique (NGOMSL) used as the starting point for design, redesign, and evaluation is able to identify quality attributes before deployment.

• Assess the value of this technique against user testing, which according to many is the most successful evaluation technique available.

• Demonstrate that the model can be used across a variety of interfaces, including hardware, ubiquitous devices and software and, most importantly to determine which usability related attributes are universal in application.

• Determine the credibility of NGOMSL and GSE as a practical means of meeting the NITRD (2003) requirements for a total system model, particularly in relation to measures of performance.

• Establish the NGOMSL-GSE method as an effective and consistent starting point for design.

It is important to note that HCI already has numerous methods for evaluating user-interface design and usability. However, they all differ in relation to time, expense, difficulty and effectiveness. To make selection a little easier the United Kingdom Usability Professional Association (UKUPA: 2003) provides a number of questions to ask when choosing a technique:

1. What usability techniques are appropriate for this project?
2. What standards will you follow and what measurements will you take?
3. What users will you test?
4. Will I get helpful and accurate answers?

This chapter details the methodology for the user studies conducted in the pilot and main studies. Section 4.2 introduces and discusses how to build and apply GSE models. Section 4.3 presents representations of the usability attributes (specified in the second chapter) and their conversion into GSE. These are used in the later studies to identify usability problems.

The remainder of chapter details how user-tests were conducted to determine the effectiveness of the GSE modelling. Section 4.4 discusses how the testing method was
chosen. Section 4.5 demonstrates how subjects were selected for each study. Section 4.6 discusses why products were chosen for the respective tests. Section 4.7 discusses the research procedures in detail, the selection of methods and outlines how test sessions were conducted. Section 4.8 documents the data analysis methods, that is, how usability problems and other data were analysed and measured. The final section 4.8 discusses the limitations of this study.

4.2 Building GSE Models

In order to test whether usability problems could be anticipated in GSE Models, in later chapters GSE Models of system and task behaviour had to be constructed. To build these models this study followed processes specified in Dromey (2002a).

Stage 1: Developing a Statement of Requirements

Stage 1 of the GSE approach is developing a statement of requirements. These requirements should be expressed in natural language and according to Dromey should be as “concise, accurate, unambiguous, consistent and complete” (Dromey, 2002a:8) as possible. This step can be skipped if an existing set of requirements can be obtained.

To demonstrate that a cognitively based model like GOMS could be used as the requirement basis for a GSE analysis, design models were built from NGOMSL specifications.

The aim was to demonstrate that GSE could easily accommodate a model capable of representing user behaviour in an explicit form. However GSE is able to use any suitable statement of requirements as the basis for development. Since execution and learning times are not calculated in this study, the question can be raised as to whether to begin with an HTA instead. After all HTA does exhibit goal-directed behaviour and hierarchical task decomposition and is an equally valid way of stating the requirements. However this study uses NGOMSL because no annotations are needed to understand the control rules, and NGOMSL is designed to accommodate cognitive operators, which, though not covered in great detail in this study, offer additional capabilities for future analysis.
Stage 2: Is perhaps the most important stage since it involves actually translating the requirements into GSE notation.

In this stage we have to identify components, states, and the other constraints, and translate them into GSE nodes. The GSE lexicon is quite large in order to accommodate a vast array of applications. However, a small subset is all that is needed to accommodate the behaviour analysed in this study. The four key GSE elements, states, if-conditions, when-states (events) and system state can be found in Figure 4-1.

<table>
<thead>
<tr>
<th>COMPONENT-STATE</th>
<th>LABEL</th>
<th>SEMANTICS</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAG COMPONENT</td>
<td>Internal State</td>
<td>Indicates that the component has realized the particular internal state. Passes control when state is realized</td>
</tr>
<tr>
<td>? IF - State?</td>
<td>IF-State</td>
<td>Indicates that the component will only pass control if If-state</td>
</tr>
<tr>
<td>? WHEN - State?</td>
<td>WHEN-State</td>
<td>Indicates that the component will only pass control when and if the event WHEN-state happens</td>
</tr>
<tr>
<td>System-Name</td>
<td>System-State</td>
<td>System-Name realizes the state &quot;State&quot; and then passes control to its output</td>
</tr>
</tbody>
</table>

Figure 4-1 The Key Behaviour Tree Elements. Wen and Dromey (2004: 105)

Translation involves taking a natural language requirement, and then formatting it to identify components and other GSE elements. Potential components can be marked in bold and potential states can be marked in square brackets (Dromey 2002a) or italics (Wen and Dromey 2004).

The next elements to be identified are the order indicators. Order indicators are:
“the events and decisions/constraints that they are associated with, the
data components exchange, and the causal, logical and temporal
dependencies associated with component interactions.”

(Wen and Dromey 2004: 2*)

These are the conditional statements and other constraints that order the structure of the behaviour tree. These may be underlined if necessary. Dromey (2002a) provides the following example of a requirement formatted for translation.

If the oven [times-out] the light and the power-tube are turned [off] and then a beeper [emits a sound] to indicate that the cooking [has finished].

This yields the tree contained in Figure 4.1.

The fundamental lexicon for GSE is quite small (see Appendix B), though many more are available if necessary (see Dromey 2002a). However only a subset of GSE notation is applied in this study. Apart from the elements already described these symbols include:

- Encapsulation @: Means the behaviour for this node is described elsewhere.
- Equivalence =: This node is equivalent to a node elsewhere in the tree also marked by the equivalence symbol.
- Iteration ^: Behaviour iterates back to an earlier node matching (the same node and state) as that containing the iteration symbol. For instance OVEN [off] ^ would return to OVEN [off]
- Concurrent Events &&: When two boxes are joined vertically and contain the double ampersand it means both states must be realized at the same time. For instance two buttons being pressed at the same time.
- Visual Output <<<>: Means data is passed to a display device. For clarity, all display nodes for GSE models in this study have been coloured green.
• Output <> Data is passed on from a component
• Input <> Data is passed into a component.

Since this study uses NGOMSL as a basis for its representations, some nodes are marked as goals (in blue) to indicate the user's high-level intentions. Further information on how to convert a NGOMSL model can be found in 5.4.

An additional issue relates to the use of iteration (looping) in a tree-like structure. Behaviour trees are a tree-like representation of system behaviour. Just like leaf nodes can be navigated in an XML document, the same applies to GSE behaviour trees.

**Stage 3: Incrementally Integrate Individual Requirements Behavior Trees to Create an Evolving Design Behavior Tree.**

The third stage of the process involves identifying how the requirement trees fit together. To do this a requirements integration table can be constructed. This lists all the requirements, their root nodes, and in which other trees, a tree's root node can be found. Using this table the tree can be slotted together, a little like a jigsaw. A detailed example of this process is documented in Appendix D.

The last two stages of the GSE process are not covered in detail in this study, since the aim is to identify usability problems early in the design process and not to analyse later stages of development.

### 4.3 Identifying attribute-based usability problems

The 22 attributes specified in Chapter 2 are used in this study to predict usability problems in an NGOMSL and GSE based design. The attributes were determined using guidelines specified by Galitz (1997), Nielsen and Shneiderman amongst others. For these to be correctly identified in GSE models, they had to be transformed into behaviour tree representations. Using GSE syntax, these attributes were built into GSE models based on descriptions in Chapter 2.
4.3.1 Applying NGOMSL with the GSE models.

This study proposes using NGOMSL alongside GSE behaviour trees. So how does NGOMSL fit into the process? NGOMSL offers a statement of requirements compliant with many aspects of GSE modelling; including a hierarchical goal-based representation of the users' anticipated behaviour. An alternative method (HTA) was discounted since it requires plans to demonstrate flow of control, and NGOMSL expresses a much broader range of behaviour.

The process for the NGOMSL to GSE conversion, which is in many ways consistent with the GSE process specified in 4.2 is described in detail in section 5.4. While the more complex elements of NGOMSL such as the memory operators, are not addressed in the method proposed in this study (though clearly this should be a goal for further research), the NGSOML method provides a statement of cognitively based task requirements from which to build GSE models. So while the approach in this study does not apply some of the more complex cognitive constructs of the GOMS family, this does not impede the primary aim of this study, which is to build a cognitively based total model of system behaviour and use predictive attributes (modelled in GSE notation) to identify usability problems.

To apply NGOMSL as the starting point for design, the following process is followed:

1. Build a NGOMSL model of the task requirements following the NGOMSL building process specified in 3.3.8.
2. Convert this model into GSE using the process described in 5.4. The NGOMSL model becomes the statement of requirements in Step 1 of the GSE building process (as outlined in 4.2).
3. Identify and eliminate usability problems using the attribute patterns in 4.3.2 to 4.3.16.

This essentially constitutes design from scratch. However a second approach is used in the studies to demonstrate the efficacy of the approach when compared against user testing:
1. Build a NGOMSL model of behaviour the same as before.
2. Build an As-Is model of the existing system behaviour in GSE based on the NGOMSL goals specified in step 1 where they apply. This is covered in 5.5 for the VCR study, and 6.3 to 6.4 in the main study
3. Identify and eliminate usability problems using the attribute patterns in 4.3.2 to 4.3.16.

These processes therefore integrate aspects of a cognitive modelling approach (NGOMSL), apply them to build a total representation of the system behaviour in GSE, and then apply GSE representations of usability attributes to predict usability problems. This approach therefore complies with the NITRD (2003) requirements for a total system model outlined in Chapter I. Now that NGOMSL and GSE have both been described, the third part of this approach it is now time to specify how the attributes themselves are employed.

### 4.3.2 Applying the Attributes

This section presents general pattern descriptions in GSE notation for the attributes identified in Chapter 2. This is a similar approach to Wood (2000) who uses GSE patterns to identify errors in GOMSL notation except that the patterns in this study are described in GSE. Based on the definition for design patterns in object orientated software design outlined in Demujian Sr and Pia (2004: 104-15) this study defines a GSE usability pattern as:

A template of a GSE behaviour tree that represents a predefined set of behaviour, which has been observed and generalised from usability guidelines and studies in order to identify new problems with similar characteristics.
It is important to clarify the role of these patterns. First of all, are they heuristic (much like Nielsen’s (2004) heuristics) or algorithmic in the sense that they could be calculated algorithmically? In a sense they are heuristic, given that they are not literal manifestations of the attributes, and rely on some judgement to determine whether the pattern applies to a given GSE model. After all, many of the usability attributes counter each other (e.g. Efficiency and Simplicity) and will need to be “traded-off” (see Galitz 1997). Nor are these patterns ready to be calculated computationally yet. It is unlikely that you will find ‘exact’ copies of these attributes in other GSE models.

However, the aim of these patterns is to provide a formal means of identifying these problems through comparison with other GSE design models. Thus the problems are represented in formal GSE notation, and therefore more quantitative in nature than Nielsen’s descriptions of his heuristics (see Nielsen 2004: 45-10). Since this is the first time a formal GSE / Usability comparison has been attempted, it is also hoped that these problems will be revised into more quantifiable forms in later studies.

Thus while, these attribute related problems are not ready to be calculated algorithmically, they are formalised enough not to be considered strictly heuristic in representation or application. They are a combination of model based evaluation (see Kieras 2003) and usability inspection, and perhaps best described as an empirical approach to finding usability problems.
Using the Patterns

The patterns in this section can be applied in three different ways. The first three reasons are similar to Wood’s (2000) arguments for using his error patterns.

1. That by applying these patterns analysts and designer may be able to prevent usability problems
2. By providing analysts with an alternative view of usability problems. To see how problems are manifested in the system-user behaviour.
3. By providing information to improve the design.
4. To better quantify usability problems, in a design context.

To apply these patterns, since no tool as yet supports them, the analyst must inspect their design using the patterns as a reference, a little like a usability inspection. This may be done at the earliest stages of GSE requirement composition, for individual requirement trees (GSE Stage 2), in later stages (GSE Stage 3 and over) as the trees are netted together in keeping with the GSE composition process or after a full model GSE behaviour tree has been constructed. The studies in Chapters 5 and 6 use full GSE behaviour trees to demonstrate how these problems may be identified in models of working systems. The technique used in these studies involved stepping through each GSE tree, scanning the trees for similar patterns as those presented in this chapter, comparing each attribute pattern in the same order as they are presented here. These are then circled and marked indicating a potential problem has been identified. The analyst then moves on to the next attribute and repeats the process for each tree in the model.
4.3.3 Forgiveness (FG) -- preventing error through input and warnings

Forgiveness is the acceptance and prevention of human error and the anticipation of user errors. This includes the assumption that input from the user may not always be correct and will make mistakes. The behaviour in Figure 4-4 demonstrates how data should be checked, and repaired if necessary, prior to any process, which depends on it.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Possible Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>FG1</td>
<td>FG1</td>
</tr>
<tr>
<td>FG2</td>
<td>FG2</td>
</tr>
<tr>
<td>FG3</td>
<td>FG3</td>
</tr>
<tr>
<td>FG4</td>
<td>FG4</td>
</tr>
<tr>
<td>FG5</td>
<td>FG5</td>
</tr>
</tbody>
</table>

 Forgiveness Problem
1. User supplies output
2. System takes input
3. System processes data
4. Process may fail due to a data problem

 Forgiveness Example
1. User supplies output
2. System takes input
3. System verifies input
4. If input is ok go to FG 6
5. If input is not ok go to FG 8
6. and 9 Data is confirmed ok
7. Correct data
8. Process Data

Figure 4-4: Forgiveness problem 1 -- unchecked data

Forgiveness is therefore a set of additional conditions added to a GSE model, after a user event or output, and a strategy for fixing the users' mistakes before they become a problem.

The first Forgiveness problem Figure 4-4 involves a component accepting data that will cause an error. In the second type of Forgiveness problem Figure 4-5 no warning message is provided to the user prior to an irreversible action.
Figure 4-5: Forgiveness problem 2 – No warning for irreversible action

In the third problem Figure 4-6 a condition exists (and is detectable) that will cause an error later on, yet the system does not warn the user and allows the action to continue.

Figure 4-6: Forgiveness problem 3 – Allow preventable error
In the final Forgiveness problem in Figure 4-7 the system sets a default value or setting, which will cause a problem later in the process.

![Forgiveness Problem Diagram](image)

**Figure 4-7: Forgiveness problem 4 – Bad default Value**

### 4.3.4 Error Recovery – return to previous state after a user error

The second attribute related to error management is Error Recovery. Error Recovery accepts that the user will make errors and will have a process defined to undo them. In design terms the system must allow the state of the system before the mistake to be restored. Figure 4-8 demonstrates how Error Recovery can appear in a behaviour tree. In this example the flow of control stops when an error occurs. Since there is no obvious recovery process this is a potential usability problem.

![Error Recovery Diagram](image)

**Figure 4-8: Error Recovery problem 1 in GSE**

Another Error Recovery problem can be found in Figure 4-9 document editing. In the left behaviour tree, the user must start all over again if errors are made. On the right
however the system is designed such that the state of the document prior to the error is retained and that the user can return to editing the document again.

![Diagram of Example Problem and Possible Solution]

**Figure 4-9: GSE representation of Error Recovery problem 2**

4.3.5 Responsiveness – immediate feedback

Responsiveness is critical to interaction, since it is associated with the feedback that a system provides. In GSE, the << >> symbols are used to indicate output to the user (as in a display). In Figure 4-10, the leftmost panel demonstrates a Responsiveness problem, where an important state change is not indicated with feedback. In the rightmost panel a solution is

![Diagram of Possible Problem and Solution]

**Figure 4-10: Responsiveness problem 1**
provided where the user outputs its response to a display component. In the example below Figure 4-11, another problem occurs when an ongoing process does not report its progress.

In this problem, the status (of the system) is displayed, in nodes RS1. The user performs an action in RS2. The process is now underway. Until the process realises state finished in RS3, it continues looping back to state [underway] via RS3r. However the status is only displayed again (in RS6) after the process is finished. Thus for the duration of the process, no feedback, and therefore no adequate response is provided. This will of course not apply to system processes that execute so quickly there is no need for ongoing feedback.

Figure 4-11: Responsiveness problem 2

In a third problem in Figure 4-12, the user is trying to enter a password. There are three possible reasons for an incorrect password, but only a single common message for each of them.

Figure 4-12: Responsiveness Problem 3
The solution is to have feedback for each type of error as in Figure 4-13.
Responsiveness is determined by the distribution of system output (feedback) for each user event.

**Figure 4-13: Responsiveness Possible Solution for Problem 2**

RS5, RS10, RS11, RS12 correspond to each possible outcome for the user entering a password (acceptance, out of date, wrong and too many attempts).

In Responsiveness problem 4 (Figure 4-14) no indication is provided as to whether a field is enabled or disabled.
In Responsiveness Problem 5, Figure 4-15 the user must move between items, displayed one at a time. The problem is that no visual cue is provided as to what comes before or after. This is especially important in navigating sequences where the next item may be inferred from the sorting.

**Figure 4-14: Responsiveness Problem 4**

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>RS1</td>
<td>DISPLAY &quot;&lt;Current Item in Last Only&gt;&quot;</td>
</tr>
<tr>
<td>RS2</td>
<td>USER &quot;&lt;?Move to Last Item?&quot;</td>
</tr>
</tbody>
</table>

**Figure 4-15: Responsiveness Problem 5**
4.3.6 Controllability (CF) -- division of control

Controllability involves the division of behaviour between system and user (the locus of control). If a system has total controllability its user must be in control at all times. In GSE this can be easily represented as a balance between user-initiated events and system-initiated events. The user should have the opportunity to cancel any system process and return to the initial system state.

A Controllability problem is relatively easy to define in GSE. In the diagram to the right (Figure 4-16), a user event is followed by a large sequence of system events, over which the user has no control. In the diagram below, the Controllability problem involves an iteration (looped) process the user is unable to cancel.

While acknowledging that some processes should not be

![Figure 4-17: Controllability Problem 2 Example](image)

1. In CT1 - Initial system state [idle]
2. In CT2 - the user initiates a process
3. In CT3 - the system changes state to the running process
4. CT4 is a check to see if the process is over. If it is over control passes back to the user in CT7.
5. In CT5 a check is made to see if the process is still going, if so it integrates back to CT3, making a loop.
6. The user is out of control in CT3, CT4, CT5, CT6.
interrupted, the example in Figure 4-17 demonstrates how a Controllability problem manifests itself in a loop and how this may be addressed in Figure 4-18.

Controllability Problem 3 in Figure 4-20 occurs when an obscure system state leads to keys being locked (and presumably functions disabled). The final Controllability example in Figure 4-19 occurs when functions are locked, or a long process is initiated due to an unexpected automatic system response.
4.3.7 Directness in GSE - [Direct visual interaction]

Directness underpins the GUI Graphical User Interface and point and click concepts. Tasks must be able to be conducted visually and directly, meaning direct manipulation of graphical entities such as cursors, icons and menu bars. The visible state of the objects manipulated by the user will change as they move the input device (e.g. the mouse).

To represent this in GSE, every significant user or system event, should lead to an external output of some kind. In GSE visual output can be represented as <<>>. The same syntax can also be used for sound components. The example below demonstrates the use of <<>> to specify visual output in a standard point and click operation.

![Diagram of Direct Manipulation Example](image)

A common Directness problem involves having to perform an atomic or simple action for an increased period of time to get any effect whatsoever. In Figure 4-22 the user has to hold buttons down for an adequate amount of time, before any action occurs. The && symbols represent simultaneous events.
Another two Directness problems relating to selecting items and activating commands is demonstrated in Figure 4-23. This diagram contains two Directness violations. The first design problem is caused by bunching up all the commands into a single command list. This includes all nodes between DR2 and DR 5. Depending on the size of this list, users may have to search for some time across a list containing information that may have been better stored in separate lists or other components, where commands could be accessed more directly. This problem excludes well-structured and intuitive menus.

The second problem in Figure 4-23 is in nodes DR6 and DR7. Instead of the user being able to enter the command using direct manipulation, they must instead enter the code using keys. Not all text entry is a violation of Directness but unnecessary text entry is a problem if no alternate direct method is available.
In Figure 4-24 fields can only be accessed in sequence, and not directly. A similar problem is described in Figure 4-25 where several other objects must be manipulated before being able to access the object the user wishes to change. The final problem in Figure 4-26 the user cannot access a value directly but must instead cycle through a long list of other values.

In the first example to the left, the user must cycle through a list of values, to get to the value they want. In the second example the user can move directly to the value they want.
4.3.8 Efficiency – goal completion according to effort

Efficiency is the rate of productivity in relation to effort. In practice this means providing shortcuts for expert users. In the example (Figure 4-27) there are two methods for printing. In the first, hotkeys are used. In the next example the menu is used. In terms of time (once time values are determined) the short cut keys are almost certain to be faster. The alternate behaviour tree specifies how these methods could be incorporated into a single GSE model, by encapsulating them into a decision structure. Having inefficient methods alongside efficient methods does not diminish the efficiency of a design, if a user may easily pick between them.

<table>
<thead>
<tr>
<th>Hotkey</th>
<th>Menu</th>
<th>Alternate</th>
</tr>
</thead>
<tbody>
<tr>
<td>EF1</td>
<td>GSO</td>
<td>EF1</td>
</tr>
<tr>
<td>EF2</td>
<td>USER</td>
<td>EF3</td>
</tr>
<tr>
<td>EF3</td>
<td>USER</td>
<td>EF4</td>
</tr>
<tr>
<td>EF4</td>
<td>USER</td>
<td>EF5</td>
</tr>
<tr>
<td>EF5</td>
<td>USER</td>
<td>EF6</td>
</tr>
<tr>
<td>EF6</td>
<td>USER</td>
<td>EF7</td>
</tr>
<tr>
<td>EF7</td>
<td>USER</td>
<td>EF8</td>
</tr>
<tr>
<td>EF8</td>
<td>USER</td>
<td>EF9</td>
</tr>
</tbody>
</table>

Figure 4-27: Efficiency example 1, short cuts for long sequences

In the second example in Figure 4-28, predictive text, a feature on mobile phones produces a larger behaviour tree than typing in the number character by character. This demonstrates that larger trees do not necessarily mean faster trees. The only way to demonstrate this is to assign times to the physical operations, and calculate how quickly an alternate design (in the right panel of Figure 4-28) generates a standard message. No user testing is needed, since the designer can use pre-defined time values.
The third Efficiency problem Figure 4-29 is perhaps the most obvious. This occurs when any significant time delay by a system component (such as a timer) prevents the user from performing their next action.

In the last Efficiency problem in Figure 4-30, the user has to move sequentially (on by one) through a set of items to choose the one they wish to use. In a long list this is very likely to be inefficient. An efficient solution to this problem is to provide shortcuts or accelerators to speed up access.
4.3.9 Learnability – Support for learning and learning time

According to Nielsen (1993) Usability is the most fundamental of attributes. There are a number of factors involved including ease of learning, time to learn, structure of information and provision and structure of learning support. Learning time can also be predicted through formulas described in Kieras (1997), if similar components are integrated after conversion into a GSE model. However, in this study, no learning time is calculated. During modelling, when support documentation did not adequately address the task, a Learnability problem was noted.

4.3.10 Predictability: Goals to Goal Achieved. No unexpected state changes

To have Predictability in a design, the user’s goals must be met, the components must work in a predictable fashion, and the system response must match the last user action. Change of state must also be carried out in an expected or natural form. The state of an encapsulated method sub-goal must produce an outcome that matches the intention of the last user event. In NGOMSL Predictability is maintained by ensuring that the outcome of an operation should be based on the goal statement. In Figure 4-31 a potential Predictability problem occurs when a file saves unexpectedly as a result of a cut and paste (although there could be an auto save feature).
In examples 2 and 3 in Figure 4-32, an unexpected state change occurs in the leftmost example. In the rightmost panel, the response depends on a state the user may be unaware of, which under some conditions may cause an unexpected response.

The final predictability problems modelled in this chapter in Figure 4-33 correspond to scrolling through a list. If the list is not sorted appropriately, or contains a range of different types of objects (which may not be logically ordered) a potential predictability problem may occur.
4.3.11 Simplicity: Minimum steps, and screen components

The logical way to maintain Simplicity is to limit complexity. Keeping the processes as close to a NGOMSL model as possible should aid in keeping methods simple.

The example to the left (Figure 4-34) demonstrates how the cut and paste procedure may be made more difficult, by adding unnecessary steps.

Another example in Figure 4-35 demonstrates how too many options can overload a single screen and how to remedy this by allocating less needed functions to a set of appropriate dialogues.
**4.3.12 Transparency – Hiding Low Level System Events**

The technical aspects of a system should not impede the interaction of the user. The user should need to understand the coding or memory requirements of an application to use it. The use of encapsulation can be used to keep the models both compact and keep low-level system functions out of the high-level interface models. The analyst is able to include all the system behaviour in the same behaviour tree but would make the trees large, cumbersome and difficult to read.
In practical terms, low-level system states that are set by the system should be encapsulated into their own behaviour trees.

4.3.13 Familiarity – Cohesion with GOMS Goals and Processes

Familiarity involves: "A natural interface, which mimics the patterns of the users’ behaviour" Galitz (1997:41). Since NGOMSL models are models of the users anticipated behaviour, Familiarity in GSE is the cohesion of a GSE model with its NGOMSL counterpart. This cohesion is demonstrated in Figure 4-38. Since NGOMSL methods contain sub-goals they can be encapsulated in GSE to maintain the different levels of behaviour.
A second potential problem exists in Figure 4-39, when an alternative process for completing a task, is significantly different to the other ways of completing it. This may not be a problem given that it is often beneficial to have multiple ways of performing tasks, for users of different experience. However significant deviations, in alternate process may prevent the user from picking it up, especially when the deviations apply to the interface, and not to any additional subtasks the user may wish to perform.

### 4.3.14 Consistency – Consistency of GSE states

Consistency in GSE is demonstrated by ensuring that system states are consistent with other system states. The left panel of Figure 4-40 shows states that apply whenever predictive text is disabled.

Whenever predictive text is in its disabled state the following must be true:

- System must be in [T9 Off] state
- The Text Mode must [be normal]
- The T9 icon must be invisible.
Thus the diagram to the right inconsistent with the conditions specified in the left panel since the I9 icon is visible but predictive text is enabled.

It is also possible to have too much Consistency. In Figure 4-41 different conditions have the same effect. This may or may not be a problem. This is the equivalent of a having a green traffic light, and a purple traffic light at an intersection, both telling the driver to go with no discernable purpose for having the second light. A better term for this problem might be redundancy. However this will be counted as a Consistency problem in this study.

4.3.15 Configurability

Configurability involves setting user preferences for interaction with the system. In the example below (Figure 4-42) the default text mode is drawn from the system to set the text mode. However the user should be able to change text mode, for the duration of the operation, or in the preferences. The user should also be able to restore defaults if necessary.
4.3.16 Flexibility -- Alternate methods for novices and experts

The final attribute in this section is Flexibility. This means that alternate methods for a single goal should be made available. These methods should also vary in terms of Efficiency and Simplicity for different user types. In the example to the right (Figure 4-43), only one method is provided to print a document. This is poor Flexibility. In the diagram below (Figure 4-44) several different methods are provided.

Figure 4-43: Low Flexibility

Figure 4-44: Flexibility
4.4 Choosing a testing methodology

This thesis has proposed a pre-deployment technique for the prevention of usability problems. In order to justify this technique, an appropriate study needs to be conducted. According to Dromey (2002a) an effective technique must meet several obligations. It must provide an advance compared to existing techniques, use and extend the best-proven ideas available, identify and overcome weaknesses in other methods and be able to identify its own limitations. But in order to be a successful usability technique, Kjeldskov and Stage (2003:610) argue that the "primary basis for evaluating a technique should be the number of usability problems it helps in identifying."

It is for this reason that in this study, usability problems are the primary criterion for measuring the success of the technique. In accordance with Dromey's (2002a) obligations for a new technique, comparison with an existing usability technique (in terms of finding these problems) becomes the primary means for determining the success of the method proposed in this study. There are several usability evaluation techniques presently available including heuristic evaluation (Nielsen 1993, Nielsen 1994b), cognitive walkthroughs (Jacobsen 1999), checklists and surveys. There are even integrated approaches such as Heuristics and User Testing in Law and Hvannberg (2002). However, in this study, user testing is chosen as the primary testing methodology, for reasons discussed in the next section.

4.4.1 User testing

The user test or usability test is chosen since it is widely perceived as the most effective way of identifying usability problems (Law and Hvannberg 2002: 72). According to Melkus (1985) usability tests remind all concerned that the user deals with a total system, consisting of hardware, software and documentation. User testing has been and continues to be validated in relation to its effectiveness (as in the CUE 1,2,3,4 studies), compared with other techniques (e.g. Jacobsen 1999, CUE 4 2003). In Jacobsen's study it is validated in relation to cognitive walkthrough, and in the recent CUE 4 study (2003) it is analysed against heuristic analysis. In both instances, the results were very favourable.
There were three reasons for using user testing. First it is the most successful method for finding usability problems, according to a number of HCI reviewers. For instance, Dumas and Redish (1999: 82) conclude that usability testing:

- Uncovers more usability problems than other methods
- Finds more global problems than other methods
- Finds more unique problems than other methods

The second reason is that in most instances it is a post-deployment technique that requires a system in a deployed state. The only exception to these conditions is if it is used in an exploratory format (see Jacobsen 1999), however all usability tests require users (Shneiderman 1998, Mehlenbacher 1995, Jacobsen 1999). Thus the third reason is that user testing requires ‘users’ unlike NGOMSL and GSE, where no users are needed. In NGOMSL users may be observed but only to determine time estimates for their actions, however many of these times are pre-determined in the literature (e.g. Card, Moran and Newell 1983).

**Conducting the user test**

Despite the fact that user testing is comprehensively documented, the setting of the study, the number of observers and the type of analysis will vary depending on the conditions of the study. According to Jacobsen (1999) there are four types of usability tests, intended for different stages of the development process as described in Chapter 2. For this study, the comparative test is chosen as the most suitable form of user testing as comparative testing compares one product against another. This study investigates alternative behaviour between systems.

This study adopted the standard format for a test, as described by Dumas and Redish (1999). They state that the common usability test situation: “consists of thinking out loud while solving tasks using the system being tested. An evaluator analyses the user’s work, and describes usability problems on the fly, or through
"analysis of a video." This format is illustrated in Figure 4-45, which demonstrates the basic testing arrangement with observer, camera, computer, desk and user.

The above arrangements are consistent with a number of recommendations including the common industry format for usability tests and recommendations provided by the United Kingdom Usability Professionals Association (UKUPA). They recommend that these tests are:

"conducted one-on-one, with an interviewer observing a user who uses the system to complete meaningful goals. It is common that users are encouraged to narrate their thoughts as they progress - usually known as "Thinking Aloud Protocol". In most cases, the observer will only work with one user at a time - not spreading their attention thinly between a large number of users" UKUPA (2003:2).

User tests must of course identify usability problems, and in order to do so must use a number of methodical techniques to gather data. The remainder of this section discusses the various parts of testing, and the different options available, in relation to ‘thinking aloud’, the role of the observer and the set up of the environment.

### 4.4.2 Think aloud and talk aloud protocols

Think aloud testing involves telling the subjects of a usability test to express their actions and thoughts by talking aloud. The users must verbalise their thoughts, feelings and opinions while they interact with the system (e.g Maidantchik, Montoni and Santos 2002: 154 and Nielsen, Clemmensen and Yssing 2002).

According to both Ebling and John (2003) and Nielsen (1994) the think aloud technique is perhaps the most valuable of all usability engineering methods. The technique documented in this study is said to be both a “think aloud” and “talk aloud” method. However Ericsson and Simon (1984), though they use the term interchangeably for most of their text, make a small distinction between the two. With talk aloud protocols subjects need only say it out loud whatever they are saying silently to themselves. In a think aloud, additional encoding is needed to elicit additional thought processes from the subjects.
This study uses “talk aloud” primarily since it addresses all six of Ericsson and Simon’s (1984) assumptions and most importantly does not “slow down task performance.” For the remainder of this chapter, talk aloud and think aloud will be used interchangeably.

Think aloud testing is a valuable means of obtaining the user’s thoughts as part of a usability study. Subjects are encouraged to think aloud, providing insight into their behaviour. According to Maidantchik, Montoni and Santos, “The Thinking Aloud Protocol method consists of compelling the test users to verbalize their thoughts, feelings, and opinions as they interact with the system” Maidantchik, Montoni and Santos (2002: 154). In addition, usability problems may also be detected through the verbalisation of the user’s thoughts.

There also are a number of other features associated with using talk aloud testing. For instance, Mehlenbacher (1993) argues that talk aloud testing:

- Identifies conceptual problems
- Uncovers problems that users might otherwise forget in interview situations
- Provides rich data from a limited pool of users
- Produces attitudinal data: data about how users feel when using the system
- Can be carried out on pieces of an unfinished prototypical system thus encouraging interactive testing.
- Necessitates an artificial testing situation due to the presence of an observer
- Does not allow measurements of time or task completion data
- Provides ‘rich’ data that are difficult to quantify
- Requires substantial time as the user must be present for each event.

Thus, Talk Aloud testing is chosen primarily so the subjects could articulate their behaviour, eliciting information that might otherwise be unavailable. This information yields valuable data, according to the assumptions in Table 4.1.
<table>
<thead>
<tr>
<th>#</th>
<th>Assumption</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The verbalisable cognitions can be described as states that correspond to the contents of STM (short term memory).</td>
<td>✓</td>
</tr>
<tr>
<td>2</td>
<td>The information vocalised is verbal encoding of the information in short term memory</td>
<td>✓</td>
</tr>
<tr>
<td>3</td>
<td>The verbalisation processes are initiated as a thought is heeded (e.g. automatic feedback)</td>
<td>✓</td>
</tr>
<tr>
<td>4</td>
<td>The verbalisation is a direct encoding of the heeded thought and reflects its structure</td>
<td>✓</td>
</tr>
<tr>
<td>5</td>
<td>Units of articulation will correspond to integrated cognitive structures</td>
<td>✓</td>
</tr>
<tr>
<td>6</td>
<td>Pauses and hesitations will be good predictors of shifts in processing and cognitive structures</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 4-1: Features of Talk Alouds adapted from Ericsson and Simon (1984:277)

4.5 Subject selection and setting

Surveys

In the studies conducted for this dissertation, surveys were used as a means of classifying participants. Surveys are condemned as a testing process in themselves (see UKUPA 2001). This is why they are often used in conjunction with other methods (e.g. Ebling and John 2003: 291). However, surveys are an acceptable means of selecting subjects, based on a number of different categorise including age, sex, and experience with video recorders.

According to Mehlenbacher (1993) there are two problems with surveys when used as the sole method for usability testing. Firstly, the users provide information after the actual experience and must recall it in order to provide accounts of any problems they may have had. The second problem is that a survey may not be representative of the population at large. In their favour, surveys are quick and inexpensive to administer, provide quantitative data and preserve the anonymity of the respondent (Mehlenbacher (1993). Since the user test is the main form of data collection, surveys were chosen for their relative ease.

In order to ensure that the tests were conducted ethically as recommended by Dumas and Redish (1999), Shneiderman (1998) Baumeister (2000), a formal record of consent was obtained.
4.5.1 Participants in the pilot study

The pilot study was conducted over a two-week period. The observation and videotaping took place in a vacant room set up as a basic usability lab according to guidelines in Dumas and Redish (1999). The room fulfilled most of their requirements for a ‘basic’ usability lab, but without access to a one-way mirror for observation.

The subjects in this study met a number of criteria:

- They all had access to a video recorder.
- They had used a video recorder at least several times a year.
- They had not used the video recorder models tested in the study.
- There were equal numbers of male and female participants.
- The study had a mix of students, professionals and industrial occupations.
- They were aged between 20 and 30 years (with a mean age of 25).
- Half claimed to know what timer recording is.
- Half claimed to have used timer recording.

According to Ridgeway (1994:2) “The attributes of the user are probably the hardest thing to control for in usability testing, as you don’t always know what characteristics are salient, nor do you necessarily want to adjust a product or its information for a particular group.”

Socio-economic background (e.g. earnings) were not covered in the study, as the prime requisite was access to a video recording device, and the sample deemed suitably representative for this study and consistent with the procedures documented in CUE (2004) and Nielsen (2004).

<table>
<thead>
<tr>
<th>Details</th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
<th>Q5</th>
<th>Q6, Q7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sex</td>
<td>Age</td>
<td>Occupation</td>
<td>Own Access</td>
<td>Brand</td>
<td>Model</td>
<td>Frequency</td>
</tr>
<tr>
<td>1</td>
<td>F</td>
<td>21</td>
<td>Retail Sales</td>
<td>Yes</td>
<td>Yes</td>
<td>Panasonic</td>
</tr>
<tr>
<td>2</td>
<td>F</td>
<td>30</td>
<td>School Teacher</td>
<td>Yes</td>
<td>Yes</td>
<td>Sony</td>
</tr>
<tr>
<td>3</td>
<td>M</td>
<td>24</td>
<td>Student</td>
<td>Yes</td>
<td>Yes</td>
<td>Panasonic</td>
</tr>
<tr>
<td>4</td>
<td>F</td>
<td>26</td>
<td>Student</td>
<td>Yes</td>
<td>Yes</td>
<td>?</td>
</tr>
<tr>
<td>5</td>
<td>M</td>
<td>22</td>
<td>Porter</td>
<td>No</td>
<td>Yes</td>
<td>JVC</td>
</tr>
<tr>
<td>6</td>
<td>M</td>
<td>27</td>
<td>Cabinet Maker</td>
<td>Yes</td>
<td>Yes</td>
<td>Canon</td>
</tr>
</tbody>
</table>

Table 4-2: Subjects for Pilot Study User Test
Participants in the main study

The main case study was conducted over a three-week period. The observation and videotaping took place in a vacant seminar room. Like the VCR study, the participants were all volunteers. The prerequisites for the study were as follows:

- Owned and used a mobile phone
- Had not used either of the two phones in the study.

All subjects in both studies were tested on a 1:1 basis. The subjects were scheduled according to their availability. Additional information on the sample in the study can be found in Chapter Six.

<table>
<thead>
<tr>
<th>Details</th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
<th>Q5</th>
<th>Q6</th>
<th>Q7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test App.</td>
<td>Description</td>
<td>Own</td>
<td>Make</td>
<td>Model</td>
<td>Network</td>
<td>How Long Owned</td>
<td>First Phony</td>
</tr>
<tr>
<td>1 M 18</td>
<td>Student</td>
<td>Yes</td>
<td>Nokia</td>
<td>3110</td>
<td>Telstra</td>
<td>More than two years</td>
<td>Yes</td>
</tr>
<tr>
<td>2 M 24</td>
<td>Animator</td>
<td>Yes</td>
<td>Nokia</td>
<td>3110</td>
<td>Vodafone</td>
<td>Not sure</td>
<td>No</td>
</tr>
<tr>
<td>3 F 22</td>
<td>School Teacher</td>
<td>Yes</td>
<td>Nokia</td>
<td>2</td>
<td>Optus</td>
<td>Less than a year</td>
<td>No</td>
</tr>
<tr>
<td>4 M 24</td>
<td>Research PhD</td>
<td>Yes</td>
<td>Ericsson</td>
<td>T20</td>
<td>Optus</td>
<td>Less than a year</td>
<td>Yes</td>
</tr>
<tr>
<td>5 F 17</td>
<td>Accountant</td>
<td>Yes</td>
<td>Stenner</td>
<td></td>
<td>Telstra</td>
<td>Less than a year</td>
<td>Yes</td>
</tr>
<tr>
<td>6 F 10</td>
<td>School Teacher</td>
<td>Yes</td>
<td>Nokia</td>
<td>21i0</td>
<td>Telstra</td>
<td>More than a year</td>
<td>No</td>
</tr>
</tbody>
</table>

Table 4.2: Subjects for Main Study User Test

Rationale for selection of subjects

Subjects were volunteers and were recruited via email and phone from a student-based population. According to Shneideman (1998: 129) “participation should always be voluntary” This is in part due to Jacobsen’s claim that “If the system tested is a cellular phone, representative user selection is almost impossible.” (Jacobsen 1999: 25) A similar approach is used in the VCR pilot study, to match the methodology of the later study as much as possible.

The pilot study used six subjects. This is in keeping with a number of studies (e.g. Nielsen 1994a, Lewis 1994, Virzi 1990, Virzi 1992) that suggest that five subjects or ‘the magic number’ is a suitable number for a test. More recently however, Spool and Schroeder (2001) argue that five testers alone may well prove nowhere near enough in many instances. The United Kingdom Usability Professionals Association (2003) also stresses that while you might not need many subjects for identifying usability problems, you will need far more if aspects of performance are measured. The main
study took these perspectives into account and, since this study includes both quantitative (error rates, task times etc), and qualitative measures (usability problems), 32 volunteer university students were recruited.

**Group allocation**
Sessions for both studies were conducted on a 1 to 1 basis, primarily for the purpose of observation. This is in accordance with the UKUPA, (2003) which recommends 1 on 1 testing to avoid spreading your attention too far around. The 1 to 1 format meant that the two video cameras could tape the activity of each subject from two different angles.

**4.5.2 The role of the facilitator and the evaluator**
Those involved in administering a user test can be assigned into two categories. The facilitator sets up and observes the test and the evaluator actually analyses the data looking for usability problems. In a study by Rowley (1994), he describes the two people administering the tests as the coach (who answers questions) and the scribe who notes down observations.

Much like user selection, there is also argument on how many evaluators to select. Hertzum, Jacobsen and Molich (2002: 663) argue that multiple users should be used to avoid what they describe as the evaluator effect. Their research concluded that a usability study should have two or more evaluators to avoid bias and find a greater number of usability problems.

In this study, the author acted as the facilitator by obtaining a room and equipment, rostering the users and observing the sessions. The data was obtained by video recording and transcribed by several different people, according to guidelines specified by the author. Two external and professional usability experts then analysed this data for usability problems across the transcriptions and using the videotapes for reference as required.

**4.5.3 The environment**
Setting up the environment is dependant on the given situation. Usability tests are often conducted in unique user testing laboratories (as in the CUE 1,2,3, and 4
studies). In other instances, analysts have taken user testing into the 'field' (Rowley 1994). The main reason for this is so the testing conditions better matched the actual situation (or context) in which the products could actually be used. However this can complicate a study, making it expensive and time consuming (Badre, Hudson, and Santos 1994: 222).

User testing environments need not be as complicated as might initially be thought. While a professional lab may use a one-way mirror (from which hidden observers may watch the study) (as in CUE 1,2,3 and 4), or sophisticated eye tracking equipment (for assessing facial characteristics) a usability lab does not necessarily need either (Dumas and Redish 1999).

In fact, in a recent study on mobile phone usage Kjeldskov and Stage (2004) concluded that more context-driven conditions using treadmills and movement, for a mobile phone study, were far less effective at finding usability problems, than traditional desk-based usability testing. They conclude as follows:

"An interesting and surprising result of our experiments is that technique 1 (sitting at a table) supports the identification of more usability problems than any other techniques. In this sense the traditionally usability evaluation technique seems superior". (Kjeldskov and Stage 2003: 614 - 615).

4.5.4 Room settings

This study conducted its tests in dedicated rooms. In the pilot study, the VCR test was conducted in a vacant room in an office complex. The room provided a large table, big enough to hold both VCRs, a TV (for the onscreen menus) and power sources for the cameras and electrical equipment.

Room setting in the main case study

In the main case study, the test was conducted in a meeting room. The room was free from interference and there was suitable room for the equipment. These test fulfilled all the requirements as specified in the common industry format for user tests. No dedicated lab (with two way mirror) was available but the test was in keeping with other usability tests operating under similar conditions.
### 4.6 Description of products

Dumas and Redish (1999: 27) argue that usability tests are suitable for all kinds of products specifically mentioning the two devices used in the study, VCRs and mobile phones.

The products in the three tests were designed to address different interfaces. Since GSE and NGOMSL can be generally applied, it is decided that only testing software would not demonstrate the flexibility of these techniques. This is why two types of hardware along with two ubiquitous devices are tested. This section documents why the VCR and mobile phone were deemed appropriate for this user study.

#### 4.6.1 Pilot study products

The items evaluated in the case study were videocassette recorders (VCRs). The core task examined is timer recording. The rationale for choosing timer recording is specified in Chapter 5, however the main reasons are that a NGOMSL model is already available and that timer recording (programming the VCR perceived to be a difficult task. Since this is a comparative usability test, two VCR models (one new and one old) were chosen, since between them there are five ways of recording a program ahead of time.
a) Formal Product Names:
The Panasonic NV-SD20 is the older of the two models. This device came with an external bar code sheet, for setting the clock, and timer recording, a remote control with an LCD panel and a scanner at the bottom of the remote for scanning in the bar codes. It had three ways of timer recording, using a panel on the VCR itself, the remote control buttons only, and the remote’s scanner and bar codes sheet.

![The Panasonic NV-SD20: Does not support G-CODE.](image)

The newer model, the Panasonic HD20 is several years newer than the SD20. The HD20 has two ways of timer recording, each of which requires access to a television, since unlike the other model, all timer recording information is displayed on a connected television. There were two ways to timer record. The first way is use the remote control and on-screen menu, to program in the details directly. The alternative is to enter G-Codes via the remote by opening the G-Code screen.

![The Panasonic NV-HD620MK2A: Supports G-CODE.](image)

Additional material
Manuals for both devices were provided, as supplied with the devices. A television guide was also made available for the G-Code task. The 5 tasks conducted on the devices are shown in Table 4-4.
### Panasonic NV-SD20 Tasks

<table>
<thead>
<tr>
<th>Task</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Timer Recording on</td>
<td>25 - 26</td>
</tr>
<tr>
<td>2 Timer Recording using the Remote Controller</td>
<td>26 - 27</td>
</tr>
<tr>
<td>3 Timer Recording using Bar Codes and Remote control</td>
<td>34 - 45</td>
</tr>
</tbody>
</table>

### Panasonic NV-HD620MK2A

<table>
<thead>
<tr>
<th>Task</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 Timer Recording (via remote)</td>
<td>24 - 25</td>
</tr>
<tr>
<td>5 G-Code Programming</td>
<td>26</td>
</tr>
</tbody>
</table>

**Table 4-4: Tasks associated with timer recording for pilot study**

### 4.6.2 Main Case Study Products: Mobile Phones

The main study focused on ubiquitous computing, where newer devices offer functionality traditionally associated with computers. Because of the prevalence of mobile phone technology and the rapid advancement in its capabilities, two mobile phone devices were chosen for the second study. These phones were attached by cable to their respective power packs at all times to prevent the battery running down. A wide range of tasks, from making calls several different ways, to taking photos and downloading games, were examined in this study.

**Formal Product Names:**

The first device is the NEC e606, which is a 3G phone (with access to a 3G network). On top of the traditional functions (such as making a call), it had an inbuilt video and single shot camera and web services. It was connected to the Three Network for video calls and web access, but for local calls used the Vodafone network. It contained a USIM, containing all the necessary details for its operation. Unlike the other device in the study, the NEC did have access to a web service provider.

The second device is the Samsung T100. This is an older phone than the NEC, but also offered web connectivity and voice-activated calls.

![Image of NEC e606 and Samsung T100](Figure 4-49: Main Study Mobile Phones)
However no Internet service provider for this phone was available in the study.

4.7 Research procedures

4.7.1 Test tasks and subjects

Specific information on the tasks and test procedures can be found in section 5.8 for the pilot study, and section 7.2 for the main study. The relevant task sheets for both tasks can be found in Appendix B. Users were provided with task sheets for both studies. In the pilot study, five tasks were used each of which corresponded to timer recording. In the main case study, two task sheets were provided with 25 tasks in total, one sheet for each of the two phones.

Subjects were instructed that they had a number of programs they wished to record. They were instructed to express their thoughts, what they were doing and any problems they might have. Subjects were allowed to ask for verbal assistance from the observer, when encountering an insurmountable problem or needing clarification. In the pilot study the subjects all began on the older Panasonic and then moved to the newer Panasonic. Each session was video taped by a single camera mounted to the left side of the user. Tasks were ticked off as they completed them. When their session finished they were asked to complete the post questionnaire, stapled to the survey and task sheet.

The main case study was conducted in a seminar room set up as a usability lab. Users were assigned a survey form, task sheet and questionnaire. Subjects were instructed that the products were being tested and not them. They were encouraged to express their thoughts and feelings. The only scenario provided to them was that they had a series of tasks they needed to complete on a new mobile phone. Half the subjects used the Samsung first and then the NEC, and the other half used the Samsung first and then the NEC. Subjects sat on a single chair, with devices on desk in front of them. User manuals were also provided. The equipment in this study included:

- Extension cord and power boards, television set (linked to TV 1).
- Two cameras, positioned at each side of the user (see section 4.5.4) each was plugged into a separate VCR, so that both could be recorded to VHS tape.
- Transformers for each phone, which ran over the table to the phones.

At intervals immediately following completion of each task, subjects were asked how difficult they perceived the task to be based on a pre-determined scale (1 very easy to 5 very hard). Five minutes was provided at the start of each of the two phone sessions to peruse the phone and manuals. The total task time was 1 – 2 hours. Forty-five minutes was allowed for the Samsung and an hour and ten minutes for the NEC.

4.7.2 Choice of the Examiners

In both studies, three examiners were used to analyse the protocols and videotaped results of the user tests. The prime requisite was that the examiners had performed comprehensive user testing in the past, and analysed the data looking for usability problems.

Examiner 1 is a post-doctoral student, with significant work experience in software engineering. Examiner 1 had conducted several usability tests (and analyses) as part of the cognitive research labs at the University of Queensland.

Examiner 2 is a professional usability expert, from interstate who has carried out several usability tests under laboratory conditions.

Examiner 3 is the author of this study, and has already conducted detailed usability tests concerning a study on the education effects of computer adventure games, as part of an honours study.

4.7.3 Protocol Transcription

In the pilot and main study, talk aloud protocols are extracted from the video taped recordings. Because of the large amount of video taped data, six Griffith University students were recruited to transcribe the task data. Each protocol corresponds to one subject for either of the two studies. Students were paid $20 per protocol. To translate the material, subjects were supplied with a list of tasks for each device as well as a button / key layout for each device which could be used to abbreviate the user's keystrokes. The task list and key layouts can be found on various pages in the Information Pack in Appendix G. Please note that only the task lists and key layouts
were provided to the transcribers, and not the additional information in the information pack, such as the attribute descriptions since only the examiners needed access to them.

A list of transcription instructions (on paper) was passed to the transcribers in person and the transcribers were able to ask instructions or clarify issues prior to the study beginning. A quick summary of these instructions for both studies is as follows:

- Dialog, comments and actions are to be transcribed and distinguished from one another on the protocols. Comments should be surrounded by quotation marks "".  
- To distinguish the subject and facilitators comments, the user’s comments can be prefixed with an initial and a colon (e.g. Z:;) and the facilitators’ comments can be prefixed with the initials (Al:).  
- Keystrokes and button presses can be abbreviated, consistent with the key layouts supplied.  
- Other actions should also be recorded. E.g. walks away from computer.  
- Manual references are of particular importance, and the section/page should be noted. E.g. refers to page 12 of manual.  
- The start and end points of tasks are to be marked into the protocols. E.g. [Task 1] etc.  
- Start and end times for each task (corresponding to the time on the VCR playing the tape) and sub tasks (as in the VCR study) should be marked accordingly. E.g. [1:01:20] – One hour one minute and twenty seconds.

### 4.8 Data analysis

For the results of a user test to be of any value, they must be analysed to determine usability problems. There are a number of different measurements that can be used to obtain information from the participants’ sessions. In Ebling and John’s (2003) user testing these measurements included time spent on each screen of the tutorial, the times to complete each exercise, the participants’ response to an exercise, the response to an evaluation survey and an analysis of the verbal protocol.
Figure 4-50 demonstrates that usability problems may be found by consulting multiple sources of data including quantitative and qualitative information. Law and Hvannburg (2002) for instance split their measures into performance measures and subjective measures. While they also use the duration of the task as a performance measure, they also examine the errors, the number of times participants seek help and the emotional expressions (e.g. frustration).

4.8.1 Finding usability problems

Ultimately the role of a user test is to identify usability problems. The data must somehow be filtered into identifiable problems. For instance Ebling and John (2000: 293) identify usability problems based on the following conditions:

- Two or more sources of evidence suggest a problem
- Two or more users suggest a problem
- One or more users crash the interface
- One or more users identify a bug in the interface, or
- One user’s evidence suggests a problem and the authors concur

Figure 4-51 demonstrates how the data, when filtered according to these conditions, does not always lead directly to identifying usability problems. A similar process is illustrated by Jacobsen (1999) in Figure 4-51. This also illustrates how data from the user sessions in the test is filtered through to become a set of unique usability problems. In the pilot study a single examiner was used. In the main study, several examiners were used. They were supplied transcripts of the sessions, and video tapes for
Determining how test data is used to find usability problems is a part of the common industry format for reporting usability problems. There are numerous studies that document how usability problems may be extracted by multiple examiners. These include the extensive CUE 1,2,3 and 4 studies.

Since these are "usability" impairing problems, they must in someway negatively affect the efficiency, satisfaction, or effectiveness of the user using the product. But not all usability problems are equal in magnitude. The next section discusses the classification of usability problems.

### 4.8.2 Classifying usability problems

Both Dumas and Redish (1999) and the CUE 4 study classify usability problems on four levels of severity. In the CUE 4 study, problems are classified as serious, critical or minor depending on their impact on the user. They are coded as P, Q and R.

<table>
<thead>
<tr>
<th>#</th>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>Positive finding</td>
<td>This approach is recommendable and should be preserved.</td>
</tr>
<tr>
<td>P</td>
<td>Minor Problem</td>
<td>Caused test participants to hesitate for a few seconds.</td>
</tr>
<tr>
<td>Q</td>
<td>Serious Problem</td>
<td>Delayed test participants in their use of the website for 1 to 5 minutes, but eventually they were able to continue. Caused occasional &quot;catastrophes&quot;.</td>
</tr>
<tr>
<td>R</td>
<td>Critical Problem</td>
<td>Caused frequent catastrophes. A catastrophe is a situation where the website &quot;wins&quot; over the test participant, i.e. a situation where the test participant cannot solve a reasonable task or where the website annoys the test participant.</td>
</tr>
<tr>
<td>A</td>
<td>Good Idea</td>
<td>A suggestion from a test participant that could lead to a significant improvement of the user experience.</td>
</tr>
<tr>
<td>T</td>
<td>Bug</td>
<td>The website works in a way that's clearly not in accordance with the design specification. This includes spelling errors, dead links, scripting errors, etc.</td>
</tr>
</tbody>
</table>

Table 4-5: Usability Categories adapted from CUE 4-Scenario (2004: 11).
Table 4-5. However it is also important to note that the CUE 4 specifications also
categorise ‘bugs’ which they identify as a violation of the design specification. This
means that there are four types of problem, which impede the usability of their
product. This is consistent with Dumas and Redish, who, as shown in Table 4-6
classify their own problems from severe to minor. The fourth type of problem, which
they note as subtle, is consistent with CUE 4s ‘Bug’ Category. This study uses
Dumas and Redish’s definitions since they are well defined and universally applicable
across all products.

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Prevent completion of a task</td>
</tr>
<tr>
<td>2</td>
<td>Create significant delay and frustration</td>
</tr>
<tr>
<td>3</td>
<td>Problems have a minor effect on usability</td>
</tr>
<tr>
<td>4</td>
<td>Problems are more subtle and often point to an enhancement that can be added in the future.</td>
</tr>
</tbody>
</table>

Table 4-6: Severity of usability problems, Dumas and Redish (1999:323-324).

4.8.3 Analysing performance measures

Besides the identification of usability problems, the user testing provides an
opportunity to collate additional data. This included:

- **Think Aloud Protocols**: Subject dialogue and comments
- **Protocol Analysis**: Task times, completion rates, number of references to
  manuals, duration of manual consultations, number of assistance requests and
  task errors (see Aleman and Vanglisti, 1994).
- **Post Questionnaires**: Phone preferences and reasons, tasks nominated as
difficult and reasons for difficulty, other problems during studies, and
recommended changes.

4.9 Summary

A short outline of the entire process, including the GSE and NGOMSL modelling is
as follows:

1. Build a NGOMSL model of the tasks, which the products must perform.
2. Build a GSE model of a system based on these tasks.
3. Build a GSE model of the As-Is system for both products using the goals and
   selection rules identified in the NGOMSL model.
4. Analyse the As-Is system model to find a set of usability problems.
5. Use the predictive properties (the attribute representations) to find usability problems.

6. Conduct a user test on each ‘AS-IS’ systems.

7. Compile usability problems from user tests according to product.

8. Compare predicted problems against the results of the usability tests.


The choice of techniques and how they are applied in the studies are summarised in Table 4-7 (over the page).

Since user testing combined with “talk aloud” is an excellent method for finding usability problems and is also a post-deployment technique as specified in the NITRD (2003) requirements, it is logical to use it as the benchmark, against predictions made by the GSE approach. The next chapter discusses these studies in greater detail, documenting the construction of a GSE and NGOMSL based model as a means of preventing usability problems. The success of this technique will be greatly dependant on just how many usability problems may well have been avoided by using it.

<table>
<thead>
<tr>
<th>Category</th>
<th>Pilot Study</th>
<th>Case Study</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-Test</td>
<td>Survey</td>
<td>Survey</td>
</tr>
<tr>
<td>Subjects</td>
<td>6</td>
<td>32</td>
</tr>
<tr>
<td>Sessions</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Duration</td>
<td>1 to 2 hours</td>
<td>1 to 2 hours</td>
</tr>
<tr>
<td>Tasks</td>
<td>5 (Six subtasks)</td>
<td>25</td>
</tr>
<tr>
<td>Method</td>
<td>Comparative User Test</td>
<td>Comparative User Test</td>
</tr>
<tr>
<td>Data Collection</td>
<td>Survey Think Aloud with Video taped protocol and observations Questionnaire</td>
<td>Survey Think Aloud with Video taped protocol and observations Questionnaire</td>
</tr>
<tr>
<td>Setting</td>
<td>Vacant room</td>
<td>Dedicated Seminar room</td>
</tr>
<tr>
<td>Cameras</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Devices</td>
<td>VCRs</td>
<td>Mobile Phones</td>
</tr>
<tr>
<td>Accessories</td>
<td>Remote controls Manuals, Bar Code sheet and TV guide</td>
<td>Chargers, Manuals</td>
</tr>
<tr>
<td>Coordinators</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Transcribers</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>Analysts</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

**Table 4-7: Comparative summary of pilot study and main study**
CHAPTER 5
PILOT STUDY
MODELLING VCR USABILITY

Although manuals contain instructions on how to operate timer recording, a lot of people either don't have the time to learn the procedure or don't feel confident in programming the device. In the case of VCRs it must be remembered that they are used by a wide cross-section of the public, many of who will not experiment with new devices. Most people wish their VCR could just tape it without user input.

(Rudolph 2001: 10)

5.1 Introduction

According to the NITRD (2003) HCI research needs, a predictive model for usability design requires a cognitive model, a 'total' system model and predictive measures for determining usability. Dromey (2003c) has demonstrated it is possible to prevent different problems with the quality of a product by applying his GSE approach to software design. However, unlike other measures related to software quality such as reliability, usability is unique in its dependence on the actions and behaviour of its user. While the behaviour of a system may be possible to predict due to control over its design, the behaviour of the user is far more difficult to anticipate. However, cognitively based models like Card et al's GOMS (1983) and a more recent variant NGOMSL (Kieras 1997) have demonstrated that it is possible to usefully anticipate many aspects of the user's behaviour by carefully analysing their goals. In order to obtain a practical outcome, this thesis proposes building NGOMSL based GSE models, and testing GSE models against attributes, modelled in GSE, that contribute to a product's usability in order to identify usability problems without access to actual users.

The aim of this chapter is to determine the effectiveness of NGOMSL and GSE, at identifying usability problems in comparison to user testing. In this pilot study, two Video Cassette Recorders (VCRs) are modelled, analysed and tested, based on GSE and NGOMSL models, and then contrasted against the results of two comparative usability tests with actual users. These results are then analysed and classified using the preventative usability attributes identified in the second chapter. Finally the
effectiveness of using GSE and NGOMSL, as a preventative approach to usability problems is discussed in relation to which usability problems could, and could not be identified in either case.

This chapter is divided into eight sections concerning the modelling and user testing conducted during the pilot study. The first section 5.2 introduces the timer recording tasks that are modelled and tested later in the chapter. Section 5.3 discusses an NGOMSL model of VCR behaviour, and section 5.4 demonstrates its transformation into GSE behaviour trees. Sections 5.6 and 5.7, document the construction of GSE behaviour trees used to model the two VCR devices used in the usability study. Section 5.8 discusses the findings of the user tests, and the usability problems found. The final section 5.9 summarises the findings of this chapter and the implications for the main study in the next chapter.

5.2 VCR Devices and Timer Programming

It might seem odd to apply GSE (a software engineering technique) to a non-software device. However, the generic nature of the behaviour tree notation is meant to be universally applicable across a range of design contexts. Thus, the functionality of a VCR device should prove to be as easily modelled as any other device, software program or web application.

This first study is not an examination of VCR behaviour in its entirety. Instead, it is primarily focused on timer programming and recording. Timer programming involves setting various values relating to the time and channel, in order to automatically record a television program later on, presumably, in the user’s absence. There are five reasons for why timer programming is considered to be an appropriate object for this first study. These reasons are as follows:

1. Timer programming is often perceived as difficult and as such is rarely used (see Ferguson 1991, Rudolph 2001, Petersen, Madsen and Kjaer 2001) Thus significant usability problems should be evident.

2. A NGOMSL model of timer programming (amongst other VCR functionality) by Keras (2002), the creator of NGOMSL, is freely available.
3. There are several different methods for timer recording, based on different interface and timer programming technologies such as On-Screen displays, G-Codes, Remote Controls and even Bar Code scanning. Each of these differs in complexity, and effort, and raises the prospect of inconsistencies such as those suggested by Thimbleby (1991) caused by the differing number of buttons between the VCR and its Remote.

4. Two VCR devices were available for the user tests (the Panasonic NV-HD20mk2a and NV-SD20 models) employing five alternate methods for timer programming across the two devices.

5. There is a common set of preconditions (such as the video being on), user input (channel, date, start time and end time), potential errors, system behaviour and task sequences underlying all five tasks.

Based on the above reasons, timer-programming tasks were chosen as an appropriate subject for GSE and NGOMSL modelling, and user testing.

5.2.1 The Timer Programming Process

The basic steps in the timer programming process are relatively straightforward. The user must first set the timer programming settings (when to start and stop recording and which channel). When the VCR’s current time, matches the recording time set by the user, the VCR begins recording. Figure 5-1 represents this process in GSE notation.

The GSE model in Figure 5-1 is not detailed enough to make any effective determination on the usability of a given device. There are far too many factors not addressed. For instance, no mention is made on how the user sets the programming settings, or how the channel is selected for
recording. This model also fails to specify if the process is applicable to multiple programs at any one time.

The usability of these devices depends on the tasks the users will perform and how they perform them. According to the GOMS rationality principle, users (for the most part) act in a rational fashion to achieve their goals, making aspects of their behaviour predictable. Since the users intentions are so important to the usability of a VCR, a natural language model of their goals is an appropriate place to begin.

5.3 An Outline of Kieras’s VCR model

Unlike the main study where a NGOMSL model is built from scratch, this pilot study uses Kieras’s (2002: 17-23) existing VCR NGOMSL model. Kieras specifies that the methods and selection rules in this model are somewhat simplified. This model is less detailed than both his text editing example (Kieras 1997) and the model built in the main study. However, this model is detailed enough to demonstrate the format of a NGOMSL model and how it may be converted into GSE notation. Kieras’s model covers a subset of full VCR functionality; namely Setting the Clock, Recording a Programme (including timer recording) and Stopping a Recording.

However since several of the important methods defined in Kieras model (such as setting a value to a value) are defined as sub-goals for functionality that is not covered by this study (e.g. setting the clock) the entire NGOMSL model is discussed and converted.

5.3.1 Assumptions

Kieras’s (2000) VCR model is dependant on several assumptions. These assumptions correspond to a set of preconditions that must be met, before any of the user’s goals can be realised. These assumptions are specified as follows:

- A cassette is loaded
- Power is on
- Desired Channel is already selected

Thus, recording is clearly not possible without a loaded cassette in the VCR. Recording is also not possible if the cassette is not writable, so the cassette must have
its protection tab intact. Most importantly of all the VCR will be effectively unusable for any purpose if it is not on. Finally, the channel to be recorded must first be specified before recording can be conducted successfully.

5.3.2 Setting the Clock

Kieras begins his model, by specifying goals and methods for setting the clock. This first goal (set the clock) demonstrates the hierarchical structure of a NGOMSL model, since it specifies a sub-goal that also specifies another sub-goal. This is how NGOMSL breaks a goal, into smaller goals and methods. The first goal is a top level or high-level goal, since it is not a sub-goal of any other method in the NGOMSL model. To set the clock, the user must set a day and time, and in order to set a day and time, the user must know how to change one value to another on the VCR.

5.3.3 Recording a Program

Unlike setting the clock, recording a program and stopping a recording are both initially dependent on a NGOMSL convention called the selection rule. A selection rule is a means of stating a user's choice between goals. Kieras suggests three different ways in which a user might record a program. Since each can be considered mutually exclusive (it is impossible to undertake one method at the same time as another) a selection rule is employed. The assumption that the user, will act rationally to complete their goals (the rationality principle), means Kieras can specify conditions believed to guide their behaviour. Where multiple goals (and methods for completing them) exist a selection rule must be stated.

5.3.4 Stopping a Recording

Though on some devices it might seem logical that only one method is needed to stop a recording, Kieras specifies different methods for each type of recording. So a selection rule is also specified here.

5.3.5 Independent Goals

There are a number of sub goals called by some of the methods in the pervious three categories. Two of these, “Set a Day and Time” and “Set a Value to a Value” are called by multiple methods.” Select a recording length” and “Cancel a Timer
Recording” are only called once. The two methods that are called multiple times, are clearly independent functions.

5.3.6 Overall Model

Figure 5.2 shows the overall structure of this NGOMSL model. Each number corresponds to one of the 13 goals. The yellow boxes indicate a method has been drafted for that goal, and a selection rule applies when the box is green.

At this stage, only the top-level goals and selection rules have been presented. To convert an NGOMSL model to a full GSE model, goals, operators, methods and selection rules (G O M S) must all be employed. The next section documents the intricacies of transforming this NGOMSL VCR model to GSE.

It is important to stress that GSE is an evolving notation demonstrated by changes in Dromeys (2002a, 2004a, 2004b). Variation between these documents includes changing notation for pre-conditions, loop representation, sub components, and parameter passing. This paper is primarily focused on the identification of usability problems in GSE behaviour trees based on NGOMSL models and usability related attributes. Additionally, like NGOMSL, GSE models are likely to differ depending on the degree of detail, and how the process is described. The GSE notation in this study is based on the three source documents already mentioned, with an emphasis on
appropriate notation to enable designers and analysts to identify usability problems. Changes are noted wherever they occur.

5.4 Converting the NGOMSL VCR model into GSE

The first stage of GSE modelling involves stating your requirements in natural language. Since the NGOMSL model contains the user’s goals (requirements) and these are already specified in ‘natural language’ this will be used as the basis of the To-Be GSE model. This section documents the transformation of the NGOMSL model into a GSE behaviour tree representation. The steps that are followed for the remainder of this section are consistent with the GSE process specified by Dromey (2002a). These five steps are in Figure 5-3.

<table>
<thead>
<tr>
<th>Figure 5-3: The Five Steps of the GSE Process</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Develop a statement of Requirements — informal, textual, possible written as conventional, functional requirements or based on use cases and their scenarios.</td>
</tr>
<tr>
<td>2. Translate each functional requirement produced in step 1 (including constraints) into its equivalent representation using one or more behaviour trees.</td>
</tr>
<tr>
<td>3. Integrate into the design, one at a time, the behaviour trees representing each functional requirement in Step 2</td>
</tr>
<tr>
<td>4. Transform the design behaviour tree to construct the component-based software architecture (CDN) for the system.</td>
</tr>
<tr>
<td>5. Project from the design behaviour tree the behaviour trees of each individual component.</td>
</tr>
</tbody>
</table>

5.4.1 GSE Step 1: Develop a statement of requirements in natural language

The first stage of GSE involves processing whatever requirements are available into a formal structure. NGOMSL has already provided us with a model containing 13 goals that correspond to user goals. These may be used as task requirements since they are tasks the user must follow to complete their tasks. The main point that a makes in relation to specifying requirements is to keep these requirements “concise, accurate, unambiguous, consistent and complete” Dromey (2002a: 8).

This study argues the point that NGOMSL is extremely well suited to GSE. The language is reasonably concise, consistent in its structure and methods, unambiguous due to its formal goal orientated structure and natural language, accurate (in so far as it is a representation of the users goals), and as complete as it needs to be depending
on the designer’s wishes. Since building an NGOMSL model is compatible with the first step of the GSE process, no other requirements are specified.

5.4.2 GSE Step 2: Translate the requirements into behaviour tree representations

The goals defined in figure 5.2 correspond to one of the 11 methods, or 2 selection rules contained in the model. These methods may be expanded into an operational sequence that includes physical operators (like press a button) and sub-goals (like Accomplish Goal: set a day and time). Most of the operators in the model are fairly simple and are not broken down into intricate detail. NGOMSL (Kieras 1997) and GSE (Dromey 2002a) give analysts the freedom to build or analyse a design at different levels of detail, depending on their needs. However, NGOMSL contains some very specific operators for modelling the user’s cognitive and physical behaviour (e.g. Card Moran and Newell 1983). The operators used by Kieras include the user’s physical operations ‘press’, ‘release’, ‘wait’ and two cognitive operators: verify and remember. Step 2 of this GSE conversion, begins with the first of Kieras’s methods to achieve the goal “set the clock”.

To keep these models as concise as possible, notation for button presses was condensed into a single behaviour box, since in GOMS and its KLM variant (Card, Moran Newell 1983) key-presses are often considered to be atomic operations and assigned a standard execution time. Key presses are unlikely to be broken down any further, although they are often accompanied with actions for locating the button and hand movement to the button. In Figure 5-4, the left most panel contains a detailed GSE representation of a simple button press. The single node is not a direct translation of the behaviour to the right. It would not be possible to translate the single node to the behaviour on the left, based on knowledge of GSE alone. In reality, the box on the right acts like an NGOMSL dummy stub, an acknowledgement that the analyst does not wish to explore this behaviour in any further detail.
It is important to also stress that the button press event excludes behaviour that involves keys that lock in, (e.g. CAPS-LOCK) or buttons that must be held for duration of time before any system response is initiated.

Starting Assumptions

The assumptions in the NGOMSL VCR model are reasonably easy to translate. There are three components each in a single state: Cassette [Loaded], Power [On], and Desired Channel [Selected]. Though these requirements are the starting point for the model, no actual mention is made of the VCR itself. The VCR is clearly the system component, it is never actually mentioned despite its importance. From a logical viewpoint, it is the VCR that must have a cassette loaded and it is the VCR that must be turned on. Another problem is whether the desired channel corresponds to the television set connected to the VCR or the VCR itself?

To address this and other issues, Dromey encourages building a 'literal translation' first, and then revising it to make it more correct. Since the VCR is the system in this model, the components are revised to “VCR [on], VCR [cassette loaded] and a Channel set to the [desired channel]. Figure 5.5 shows the original literal translation and the revised translation. Because the VCR is a system component it has the double line border. The VCR [Cassette [Loaded]] is a binary relation that indicates the VCR has the component Cassette in a loaded state. The CHANNEL component is specified as a sub-component of the VCR corresponding to the currently active VCR Channel.
In Figure 5-5 each of these assumptions have been tagged with an “0+”. The 0 is used in this thesis as a means of indicating these assumptions precede the 11 goals and 2 selection rules in the user requirements. The + indicates that the three component states are specified preconditions in accordance with GSE notation (Drome 2002a). There is also “-” notation for implied preconditions that are not directly stated in the requirements.

With the assumptions modelled, the next step is to analyse the goals, methods and selection rules. Each goal is numbered from 1 to 13 (since 13 goals in the model). The first goal (goal number 1) and its corresponding method involve setting the clock.

**Goal 1 Method: Setting the Clock in NGOMSL**

The first goal in Kicas’s model is to set the clock. In its original NGOMSL form the method drafted to achieve it is as follows:

**Method for Goal: Set Clock**

Step 1. Press CLOCK button
Step 2. Accomplish Goal: Set a day and time with the current day and time
Step 3. Press CLOCK button
Step 4. Return with goal accomplished

This notation, thanks to its natural language, is straightforward. The first unnumbered line indicates that by executing this method it will allow the user to achieve the goal “Set the Clock”. The first line is an operator, a physical step in the execution of a method. The second line is a sub-goal, a goal that must be achieved to complete the
current goal of setting the clock. In other words, to set a VCR clock, one of the steps involves setting the day and time, which like all the other methods will have its own steps. The final step in the Set Clock Method specifies that when the clock button is pushed one more time, the final step of this method has been accomplished, and the goal achieved.

With the goal and method specified it is now possible to convert these steps into GSE. In accordance with the GSE process, the analysis begins by identifying potential components and potential states. After looking for nouns (components) and verbs (actions) in this first method the NGOMSL notation will be formatted accordingly. In the following Set Clock Method potential components are in bold. Potential states are marked by square brackets. Underlined sections indicate the goal name as well as any sub-goals called by the method.

**Method for Goal: Set Clock**

1. [Press] CLOCK button
2. **Accomplish Goal:** [Set] a day and time with the [current day] and [time]
3. [Press] CLOCK button
4. [Return] with goal [accomplished]

The major problem is that this requirement also contains specific NGOMSL terminology such as goal, method and “accomplish goal”, which are not physical components with which the user interacts. Secondly, no explicit mention of the user is ever made in this method. Instead, the NGOMSL sequence is written like a tutorial (e.g. do this, then this, then this) to keep it more natural. To make more sense of the method, this study suggests the following steps, for formatting the NGOMSL steps into a GSE compatible format.

1. **Begin by converting the first step into a goal event**
   User has a **Goal**, which when realised, initiates subsequent behaviour in the tree.

2. **Convert the last step into a completion state for the goal**
   The User’s Immediate **Goal** has been [accomplished]

3. **Establish the User’s presence as a component in all operators.**
   Since every step in a NGOMSL model, is representative of the user’s behaviour, assume the user initiates each operation in the NGOMSL sequence. Thus a User Component exists for all steps in the process, however since Goal, already implies the users presence, no user statement is required for “goal accomplished”
4. **Distinguish Components and States**
   a. Components are almost always nouns, but not all nouns are necessarily components. Also where one thing is set to another in a step, the first entity is usually a component and whatever it becomes is a state. Components can be marked in bold.
   b. States are typically verbs, in past or present tense. Prepositions (a, the, with, for etc) can usually be ignored. States can be marked in brackets.

5. **Classify Conditional States**
   Some states will be conditions. Look for the presence of an IF or else statement prior to a marked state. If user wants to mark them prior to conversion round brackets can be used.

6. **Mark Sub-Goals**
   The “accomplish goal” statement documents the existence of a sub goal. Use an Underline to mark these sub-goals.

After applying this process (and removing unnecessary “step” prefixes), the formatted NGOMSL notation now looks like this:

```
GOAL: [Set the Clock]
1 USER [Press] CLOCK BUTTON
2 Achieve goal: set a day and time with the [current day] and [current time]
3 USER [Press] CLOCK BUTTON
4 Return with goal accomplished CLOCK [set]
```

It is now possible to build a literal GSE model of the first method. The method begins with an event, based on the state of the user’s present goal. The method will only be triggered, when the user’s immediate goal becomes “setting the clock”. For modelling purposes, The GOAL component can be considered to be a USER component also, but is marked as a GOAL, to keep the NGOMSL notation recognisable. In keeping with GSE conventions this event is represented with double question marks.

The first numbered step of the Set Clock sequence is a physical operation performed by the user. This can also be considered to be a NGOMSL operator. In this instance, the user presses the clock button. This action is also an event, and is therefore marked using the GSE
"??" symbols. To keep the models as concise as possible, this step uses the simple key press event mentioned in Figure 5-4, and is not broken into further detail.

The second numbered step in the Set Clock method is a sub goal. This occurs anytime the NGOMSL “accomplish goal” statement is used. Since the method for this goal exists elsewhere, the encapsulation symbol @ is used indicating that this behaviour is modelled in another behaviour tree. As in the previous goal statement (Figure 5-6), sub-goals are also marked in blue boxes.

When this goal is completed, the method specifies that the time and day will be set to current values. To keep the model as complete as necessary, additional boxes can be used to document this change in state.

The final step of the clock set method, specifies that the goal has been achieved and the clock is now set. This means two boxes are needed. The first to indicate that goal has been accomplished, and another to represent the state of the clock.
This entire process is put together in a literal form in Figure 5-12. While this process seems perfectly suitable at a high level of detail, it does little to demonstrate data flow (how the user enters the data in, and how this information is passed on to the day and time components.

In a second revised variation in Figure 5-13 the user passes on values for the current day and current time to the day and time components.

There are some new GSE entities in the revised version. In 1.3 a “screen” component is added in accordance with GSE screen notation (as in section 13 Dromey 2004b). Since the user must enter the Current Day and Current Time values and the system must provide some visual output as to the state of this data a screen component is appropriate. The << and >> symbols signify that this display outputs the values to the screen, and the >> and << mean that the user sets these values to the screen. For the remainder of the thesis these screen components are displayed in green.

Setting these values is accomplished by an event (the sub goal) set day and time (so it is marked in blue). The sub goal, can be contained in a separate container, but in this instance is linked to the last part of the display component, to demonstrate that it is accomplished here.
The encapsulation @ symbol indicates that the steps in this sub goal are defined elsewhere. The < and > symbol indicates that the Current Day and Current Time values are passed on as output to the DAY and TIME components after the set day and time event is initiated. When this happens the clock is set, and the goal is complete. Now that this representation is complete, the next step in building a GSE translation of this model is to define the Set Day and Time method. Since the encapsulation symbol specifies that this process is defined in a separate method.

**Goal 2 Method: Set a Day and Time**

The second method is provided to accomplish the Set a Day and Time sub goal specified in the set clock method. In this method, the hours, days and minutes for the clock are now changed, by passing them on to a sub goal.

**Method for goal: [set] a Day and Time**

1. Accomplish goal set day value to [desired value]
2. **User** Press Select
3. Accomplish goal set hour value to the [desired value]
4. **User** Press Select
5. Accomplish goal set minute value to the [desired value]
6. Return with goal [accomplished] - Day is now [set] and Time is now [set]

The first box corresponds to the user’s goal “Set a Day and Time”. The second step calls the sub-goal for setting a day to a desired value. This sub goal is (set a value to a value) is also used to change the hour and minute. Each of these sub-goals is followed by an immediate change in state.

Set a Day and Time is at a lower level of detail than that in the set clock models. In the revised version in Figure 5-14, attribute notation (marked in lower case and with a: prefix) is introduced to better demonstrate how these values are actually set. These attributes are actual values corresponding to their respective components, and while nor having states of their own, can be set to values using standard operators such as + = - and so on. Component DAY now contains a day value attribute, and Component TIME a minute value and an hour value. Desired Attributes for these values are inputted in 2.1, 2.3 and 2.5. Since Kieras has specified an additional level of detail for entering values, the encapsulation symbol is used one more time in the set clock methods.
Because the "Set a Value to a Value" method is generic and applies to all three values, it is important to indicate that the same method applies to each, and is repeated three times, with a different value each time. In this study the # arbitrary value is used (in Dromey 2004b, 7.3). The Set a Value#Day:value to DesiredValue#Desired Day indicates that two parameters are passed to this sub method. The first is the attribute value which will be changed, and in each instance (2.1, 2.3 and 2.5) it is specified as Day:DayValue, Time:HourValue and Time:Minute Value respectively. This value is followed in each declaration by the corresponding desired values (the new value) which are DesiredDay, DesiredHour and DesiredMinute Values. The @ encapsulation simple, indicates that this is processed in a sub method.

This may seem a little complicated, but at the time of writing, a complete and final set of attribute notation is not yet formally specified in the documentation (i.e. Dromey 2004b). The above notation is based on the arbitrary notation in Dromey (2004b) and
is likely to be updated soon. However, this notation maintains as much of the original
NGOMSL model (such as the goal: "Set a value to a desired value") as is possible
while demonstrating that this method is used three times and with different values.
The # symbol may be found in nodes: 2.8, 5.1, 6.14, and 6.15 in Appendix A, to
represent an arbitrary element or type. The designer or analyst depending on their
required level of detail may choose a more simple translation, as in the literal version
in Figure 5-14. However the arbitrary notation is an effective way to specify data
passed on to sub-methods and the subsequent change in value.

**Goal 3 Method: Set a value to a desired value**

Part of the NGOMSL approach is to ensure consistency. By drafting a generic method
for setting the DAY, HOUR and MINUTE values, a consistent and common set of
operations can be specified for each value.

**Method for goal: [set a value to a desired value]**
1. **User** Verify that display shows [current value is flashing]
2. **User** Decide: If display (shows value [same as desired]), then return with goal
accomplished
3. **User** Decide: If display (shows value [smaller than desired]), then [press] UP button.
4. Return with Goal [Accomplished]: Value [set to desired value]

---

**Figure 5-15: GSE Representation of Goal 3: Set a Value to a Desired Value**
The absence of “Achieve Goal” statements indicates that no sub goals are used in this method. Instead, it relies entirely on operators. In NGOMSL, high-level operators can be broken down into atomic operators. For example pressing the up button, could be broken down into the user finding the key, moving their hands to it and then pressing it. However, this NGOMSL model does not go into that kind of detail since that is typically the role of the keystroke level model (another variant of GOMS) discussed in chapter 3.

There are two mental operators in this method. The ‘VERIFY’ (3.1) and ‘DECIDE’ (3.2) operators are considered to be mental operators, because they describe the user’s cognitive behaviour. Kieras (1997: 739) defines VERIFY as a primitive external operator and says it corresponds to the M (mental operator) in the KLM syntax. DECIDE is considered to be a standard primitive mental operator, for specifying the flow of control (Kieras 1997: 739 – 740).

This method also requires a choice between events. When events 3.3, 3.4 and 3.5 are arranged as they are here only one event may be followed. Which event applies is determined by which is found to be true. Each event is mutually exclusive, that is, the user may only make one of these decisions. Single conditional nodes are not used here, since flow of control is not automatic. Control continues only after a user makes a decision, on whether the Displayed value (DISPLAY: Value) is equal, less than or greater than the Desired Value they wish to set (desiredValue: Value).

Another important feature of this method is the use of iteration and looping (marked as the original 3.1 node and the two 3,1R nodes). If the value does not match the desired value, the value is decremented or incremented, and the whole process is repeated, until the user gets the value they want. This is accomplished by using the symbol, indicating that after changing the value, control iterates back to the closest previous node with the same behaviour. In this case, the procedure is iterated back to the user checking the value again.

This concludes the methods that Kieras assigns to setting the clock. However, both sub methods, setting the day and time, and setting a value to a value are called by the
timer recording methods later in this model. The next section concerns methods relevant to recording a program.

**Goal 4 Selection Rule: Recording a Program**

Having already seen Goals, Methods and Operators (three parts of the GOMS title) goal number 4 introduces the last of the primary GOMS objects: the selection rule. Selection rules are a critical part of NGOMSL modelling, and are used to represent the decision-making (by the user) between alternate methods. In a NGOMSL model the user meets their goals, by choosing between methods based on specified conditions specified in the selection rule. When formatting this selection rule, it was not necessary to add a reference to the user since this selection rule is written in the second person. For the purposes of translation each “you” in the selection rule can be replaced with “user.” The formatted version of a selection rule is as follows:

**Selection rule set for goal: [record] a program**

1. **If (you are present when the program [starts]) and (you will be [present] when the program [ends]) then accomplish goal: [record] a program [manually]**
2. **If (you are [present] when the program [starts]) and (you will [not be present] when the program [ends]) and (you [know how long the program lasts]) and (the VCR clock is [set]) then accomplish goal: [record] a program with [One-Touch Recording]**
3. **If (you [will not be present when the program starts]) and (you [know when the program will start]) and (you [know how long the program lasts] and the VCR clock is [set]) then accomplish goal: [record] a program with Timer Recording**
4. **Return with goal [accomplished] – program [recorded]**

![Figure 5-16 Selection Rule Set for Goal Record a Program](image-url)
The formatted selection rule, contains several conditional statements chained together by the "IF" and "AND" statements, leading to three possible outcomes. In order to represent each IF sequence, the conditionals are stacked one on top of another and marked with a "&" symbol meaning the adjoining boxes are part of an "and" condition. Control is only passed on if all the "and" conditions are satisfied.

**Goal 5 Method: Record a Program Manually:**

The first method for performing a record is to do so manually. When the program starts the user hits the record and play buttons. When it stops they hit the stop buttons. The formatted NGOMSL sequence is as follows:

**Method for goal: [record] a program manually**

1. User [Wait] for program to [start].
2. User [Hold down] REC button.
4. User [Release] both buttons. *Note: PLAY and REC*
5. User [Verify] that display [shows "REC"] and arrow [is moving]
6. User [Wait] for program to [end]
8. Return with goal [accomplished] – Program [recorded]

The only difficulty in documenting this behaviour is that two buttons (Press and Record) must be pressed simultaneously to begin the recording. In order to represent this, GSE notation for conjunctive transfer of control was used. Unlike the selection rule, the ampersands were used to mark pressing the play button and holding the record button into a single condition.

![Diagram of selection rule set for goal record a program manually]

*Figure 5-17 Selection Rule Set for Goal Record a Program Manually*
Goal 6 Method: Recording a Program with One Touch Recording

One touch recording is a way of specifying how long a program will tape. This technique is a rapid alternative to timer recording, however the user must be present at the start of the recording in order to activate and set the duration of the recording. The user simply hits the OTR button repeatedly until the recording time is large enough to record the program.

Method for goal: record a program with One-Touch Recording
1. User [Wait] for program to [start]
4. User [Decide]: If time shown in display (is less than length of program), go to Step 3.
5. User Return with goal [accomplished]. - Program [Recorded]

Goal 7 Method: Recording a Program with Timer Recording

The final recording method specified in the selection rule (Goal 6) is timer recording. This is the most important of the methods in this study, since the usability test later in the study analyses several timer-recording methods. Timer recording requires the most input from the user. This makes it more complicated than either one-touch or manual recording methods. This is demonstrated by the larger number of steps needed to achieve this goal, the presence of a new sub-goal (Goal 8-Set the recording length) and reference to other sub-goals used to set day and time values already described as part of the set clock operation in methods for goals 2 and 3. The formatted version of timer recording is as follows:
Method for goal: [record] a program with [Timer Recording]

1. User [Press] PROGRAM button
2. User [Verify] that program number [flashes]
3. User [Press] SELECT button
4. Accomplish goal: [set] a day and time with the [starting day] and [time] of the program
5. User [Press] SELECT button
6. User Accomplish goal: [select] recording length
7. User [Press] SELECT button
8. Accomplish goal: [set] channel value to [desired value]
10. User [Verify] the display returns to [normal display]
11. Return with Goal [Accomplished] – Program [recorded]

The literal version of this NGOMSL model is contained in the following diagram:

![Diagram](image-url)

Figure 5-19: Timer Recording in GSE (Compressed vertically to fit on page)
Goal 8 Method: Set the Recording Length

Setting the recording length is the only value not set using the Goal 3 method (Set a Value to a Value). Instead Kieras chooses to draft the entire method as a set of operators with no sub goals. This method loops until the user is satisfied with the value shown on the display.

Method for goal: [select] the recording length
Only certain values are allowed, so special method required

1. User [Verify] that display [shows "LGTH"]
2. Press UP button.
3. User [Decide]: if time [shown] in display is less than length of program, go to Step 2
4. Return with goal [accomplished]: recording length [selected]

Like an earlier goal, the time is assumed to have incremented each time the UP BUTTON is pressed, and then passes this value on to the display. Determining the location and purpose of these system components is important at latter stages of the GSE process, particularly stage 5, where the full functionality of the system is determined.

Goal 9 Selection Rule: Stopping an Ongoing Recording

Like Goal 4, Goal 9 is structured as a selection rule. This selection rule is similar to Goal 4 in the sense that it has three ‘and’ conditions for choosing between sub goals. Unlike Goal 4 however the sub goals correspond to stopping a recording.

Kieras acknowledges that there are elements on the display, which could be used as alternate conditions for picking a method. However the selection rule for goal 9 specifies the cognitive operator “remember,” using it as the means of choosing
between alternate select statements. Thus, instead of conditions, nodes 9.1, 9.2, 9.3 are mutually exclusive events instead. The formatted version of this goal statement is as follows:

**Selection rule set for goal: user] stop an on-going recording**

There are subtle cues on the display that could also be used to select the method

1. **If you** [remember] that the **recording** is [manual] then accomplish goal: stop an on-going manual recording
2. **If you** remember that the **recording** is a [One-Touch Recording] then accomplish goal: stop a [One-Touch Recording]
3. **If you** [remember] that the **recording** is a [Timer Recording] then accomplish goal: stop a Timer Recording
4. Return with goal accomplished.

![Diagram](image)

**Figure 5-21: Selection Rule Stop an ongoing recording**

Ultimately, all three options serve a common purpose; to stop the VCR recording. In order to represent this in GSE, the system component (the VCR) was repeated three times in the [recording stopped] state. The equal sign has been used in keeping with Dromey's (2002a: 42) recommendations in order to preserve the CDN Component Data Network built in stage 5 of the GSE process.
Goal 10 Method: Stop an Ongoing Manual Recording

Method for goal: (User) [stop] an [ongoing] manual recording
1. User Press the STOP button
2. User [Verify] that "REC" [disappears] and arrow [stops moving]
3. User Return with goal [accomplished]: recording – [stopped]

Stopping a manual recording is one of the easier methods, and uses a conjunctive condition for the user to verify the state of the REC and Arrow entities. The visual nature of these entities suggests that they are part of the Display, and thus no separate components were provided for "arrow" and "rec.". This method ends much like the selection rule for goal 9, with the VCR (the system component) ceasing recording. Like other methods in this NGOMSL model no errors are specified.

Goal 11 Method: Stop a one touch recording

Stopping a one-touch recording is slightly more complicated than stopping a manual recording. In Kieras's model an iterative solution is specified, where a user must press the OTR button until the time is not greater than 0 (meaning no more recording time)

The formatted version of this NGOMSL method is as follows:

Method: [Stop] a One-Touch Recording
Depends on time delay since OTR button last pressed,
but this method always works
1  User [Press] OTR button.
2  User If display (shows a time greater than zero), go to Step 1
3  User [Wait several seconds]
4  User [Verify] that VCR [turns off]
5  User Return with goal accomplished
Goal 12 Method: Stop a Timer recording

Stopping a timer recording is an unusual process. First of all the power button must be pushed. This is presumed, not to turn on or off the power, but to end the timer-recording mode on the VCR. The formatted version of this method is as follows:

Method for goal: User [stop] a Timer Recording
1 User Press POWER
2 User [Verify] that display [shows "REC" disappears] and arrow [stops moving]
3 User Accomplish goal: [cancel a Timer Recording]
4 User Press POWER
5. Return with Goal Accomplished – Timer Recording [stopped]

No operation for cancelling the timer recording procedure is specified in the above method. Instead Kieras chooses to pass this procedure on to a sub-goal (goal 13) instead.

Goal 13 Cancel a Timer Recording

This final method is a called by Goal 12 and details the specific set of operations involved in physically cancelling timer recording, eventually returning the VCR to its original state (though this particular system state is not documented in this method). The method for Goal 13 is primarily a set of operators as steps 1,2,4,6 are all physical operators (e.g. press a button). These are not broken down into any more detail. This method is at a third level of detail for the entire NGOMSL model with “stop an ongoing recording” at level 1, and “stop a timer recording at level 2”. Since the majority of this method consists of operators, a number of physical components are identified in this method, including several buttons. Using the translation rules applied to the previous methods, the formatted method and its GSE representation are as follows:
5.5 GSE Models of the VCR devices

Unlike the NGOMSL-GSE model of VCR operation in the first section, the next two GSE models are based on the behaviour of the two VCR devices used in the study. Only behaviour relevant to the timer programming tasks covered in the user tests, are modelled. These models cover four task-related goals:

1. Setting a timer recording
2. Checking a timer recording
3. Correcting a time recording
4. Cancelling a timer recording.

Two sources of information are used to build these models. First and foremost the official user manuals for each device and the instructions in them for timer programming on each model are consulted. Secondly, the analyst also examined the devices, by following the tasks to obtain additional information on the systems behaviour. This enabled the analyst to accurately represent the capabilities of the device, and obtain additional information when it was needed.
Formatting the manual sequences

In order to make use of NGOMSL modelling principles similar goals, and operators have been integrated into the As-Is model of the device where possible. The following sequence is formatted in a similar manner as that used to convert the NGOMSL goals in 5.4. This formatting aims to preserve the natural language where possible as well as identifying the components and states. Thus process is used in both studies. The aim is to produce GSE models, capable of being compared against formal representations of the usability attribute problems in Chapter 4.

Timer Recording Data

As in the NGOMSL model similar data is needed to set a timer recording. This data includes:

1. **The Program Position (Channel).** The Program position on both devices may be set to a *standard channel* represented by a numeric value between 0 and 99, or an *external signal* such as auxiliary port AV1.

2. The **Date.** This can be a:
   - a normal numeric date. This is a value between 1 and 31 depending on the days in the month. The VCRs automatically assign any standard date value, before the current day as a value corresponding to the following month. Thus a standard date consists of days from the current date to the end of the current month, and days from the start of the following month to exactly one month on from the current date.
   - a Weekly Recording Value. This is when a timer recording is activated on a specified day every week. The displayed values are Su, Mo, Tu, We, Th, Fr, Sa
   - a Daily Recording Value. A range of days in which a timer recording will be initiated for each of those days. There are three ranges Su~Sa, Mo~Sa and Mo~Fri]

3. **The Starting Time** of the recording. This includes the **Starting Hour**, a numeric value between 0 and 23 and the **Starting Minute** between 0 and 59.

4. **The Ending Time** has an **Ending Hour** and **Ending minute** similar to the starting time.
Unlike the NGOMSL model, the two VCR models use an ending time, instead of a taping duration. Put simply, instead of saying record for two hours, the user specifies an exact time at which the recording should stop.

An additional difference is that the two VCR models used in the study, allow the user to set multiple timer recordings. The VCR devices support 8 timer-recording programs. This means that the user can schedule up to 8 timer programs (with different dates, channels, starting and ending times. This means a fifth piece of information is needed to determine the time recording, to numerate and identify the programs. This value is simply referenced as the program number or index. Using these values, both VCR models are able to accurately record the users timer recordings.

Even though each model uses the same information, the procedures for entering this information differ significantly between them. The next two sections document the construction of these GSE models for both the NV-SD20 and HD-620 VCR models. The purpose of these sections is to build GSE models of the Timer Recording procedures based on the actual devices using an NGOMSL goal structure similar to that used in Kieras’s NGOMSL model. These models can then be analysed against the attribute behaviour trees modelled in Chapter 4, to determine which usability problems could be identified, and to what extent a NGOMSL based GSE model can be used to find these potential problems. These can then be compared against the user testing results to determine how successfully potential usability problems can be identified without access to actual users.

5.5.1 Representing GSE models
Due to the considerable size of GSE models later in the chapter, many of the larger behaviour trees are not represented in the natural top down form displayed in Dromey (2002a and 2004b). Instead flow-lines are directed according to best fit, as demonstrated in Figure 5-26.
5.5.2 Representing Attribute Violations

In order to identify problems with the attributes in question, the GSE models are examined against the example problems for each of the attributes represented as GSE diagrams in Chapter 4. To find these problems the models of the goals and tasks in this chapter are scanned sequentially against the attribute diagrams in Chapter 4. The analyst does manually this since no automated tool is yet available. Whenever a potential problem is identified it is recorded and described alongside the problem diagram in Chapter 4 to which it corresponds.

5.6 The Panasonic NV-SD20 Timer Programming model

The Panasonic NV-SD20 is the older of the two VCR models used in the study and corresponds to tasks 3 –5 in the pilot study user tests. Unlike the other model, the NV-SD20 has three different ways of timer recording. Two of these methods rely on the remote control. The other method relies on buttons that are fixed to the VCR device itself.

5.6.1 Timer Recording Methods – Description

The SD20 has three methods for timer recording. These are as follows:
1. Using the VCR only (no remote is necessary). [Task 3 in study]
2. Using the Remote Control only (without the bar code sheet). [Task 4 in study]
3. Using the bar code reader on the remote control, and the coding sheet supplied with the video recorder. [Task 5 in study]

In the first of the methods the user must operate a panel of buttons on the actual VCR device. Feedback is provided by an electrical display, not unlike the standard alarm clock.

SD20 – Timer Recording using the VCR Panel

The entire timer recording procedure on the SD20 can be executed without touching the remote control. This involves using buttons on the front panel of the VCR, and switching between values presented on a small electronic display. This display shows all the numbers e.g. (channel, date start-time and end-time,) needed to set a timer recording program. The VCR device supports eight timer-recording programs, meaning that the user can schedule up to eight programs (including different channels, dates and times) for timer recording.

The timer record values are shown in the bottom part of Figure 5-27. The user navigates between them by pressing the NEXT or BACK buttons (as shown in the top part of Figure 5-27), until the correct item starts flashing indicating its readiness. Setting values for each of the timer recording items involves clicking the + and – buttons to decrement or increment the value.

Four of the buttons on the VCR
panel have duel functionality by changing between Minus or Down, Plus or Up, Next or Speed, and Shift or Sleep modes. Lights above these buttons show are lit up to indicate which mode the button is actually in.

**SD20 – Timer Record Technique 2: Using the Remote Control**

Unlike the first timer recording method, the second relies upon the remote control. One of the most significant differences between the SD20 remote, and that used by the other model is the inclusion of a LCD screen.

![Image of VCR Timer Record Layout](image)

*Figure 5-28: VCR – Timer Record Layout*

This LCD screen (as shown in Figure 5-28) can be used by the user to enter, and check timer recording values. These values are not automatically sent to the VCR as they are entered. The user is able to enter the settings and then choose to transmit them to VCR by pressing a transmit button on the remote. This is the first of two-timer recording methods using the remote. The third and final method requires using a barcode reader.
SD20 – Timer Record Technique 3: Using the Bar Code Sheet

At the bottom left corner of the SD20 Remote, lies a scanning device, which is activated by pressing the SCANNER ON/OFF button on the remote. This procedure involves the user swiping the scanner across codes supplied on a sheet of bar codes supplied with the recorder. These sheets include codes for activating timer recording, and setting the clock. Once these functions are activated, the sheets can be scanned for channels, start times, end times, and tape speed. Like the other remote technique, the scanned items also show up on the LCD display. The user must also press the transmit button, if they wish to commit the items to the main VCR unit.

5.6.2 Using the VCR Panel

The first method modelled is that for timer recording from the VCR. This has been adapted from page 24 of the Panasonic NV-SD20 Operating Instructions Manual, and by observing its actual operating behaviour. The formatted text used as the basis for these models is as follows:

Checking / Selecting a Timer Program -- Formatted Description
To check or select a program the VCR must be [on] OR in [Timer Record Mode]. The user must then [press] the Program Button. If the clock is [not set], the VCR will [beep] and nothing else will happen. If the clock is [set] the VCR will display the current record, in the Timer Programmes list by outputting its values Prog Position (channel), Date, Starting time hour, Starting time Minute, Prog Position, Date, Starting Time Hour, Starting Time Minute, Ending Time Hour, and Ending Time minute, to the screen. The VCR PANEL is now in [Timer Recording]
mode and accepts this data as input for each of the above values to its own fields and displays them to the screen. If the VCR is in (Timer Record Mode), the VCR Panel is [disabled] and the only thing the User can do is [check another timer recording] by [pressing] the PROGRAM button again, repeating this process. If the VCR (isn’t in Timer Record Mode), the user may now, achieve goals, check/select another program, set another timer programme items, or cancel the current timer program

To Check another Timer Programme
The user needs to [press] the CHECK PROG Button, again returning to the start of the process

Cancelling a Timer Programme
To [cancel] a timer programme the user must [press and hold] the + and – Buttons simultaneously [for 3 seconds]. This will [delete] the [current] record in the list of Timer Programmes and the VCR panel will then [clear] all its values.

To set the Timer Record Values and activate timer recording
The user must set the timer programming values before [activating timer recording] by [pressing] the TIMER REC button. The VCR will either enter [Timer Record Mode] or an error will occur and the VCR will beep

The GSE conversion of this model can be found in Figure 5-30. This model contains only the behaviour defined in the supporting material, and observation of the device itself. It is now possible to assess this model according to the attributes specified and described in chapter 4 in order to find potential problems.
Attribute Analysis: for Timer Recording with VCR Panel Part 1:

The VCR panel exhibits Simplicity, since no more than what its needed is displayed to the panel, and Visual Appearance and Layout since it is displayed in the order it will be entered. Transparency is present, but could well be a negative, since the device reveals very little about its state. Familiarity cannot be determined yet, although it is roughly similar with the users goals. Flexibility is an issue, since no alternate methods is actually specified anywhere.
Potential problems in Figure 5-30 are as follows:

- **Problem V1**: If the clock is not set, the user cannot proceed to timer recording. The VCR beeps in response to indicate an error.
  - **Responsiveness**: The VCR Beep is the only response to the Clock not being set which is hardly an informative response. This problem corresponds to nodes RS4, RS6, RS7, RS8 in Figure 4-12
  - **Error Recovery**: The situation can be remedied but only by the user accessing the set clock functions, but the system provides no support other than the VCR’s beep. This problem corresponds to the sequence beginning with node ER2 in Figure 4-8
  - ** Forgiveness**: No advance warning is provided that the clock is not set. Refer to nodes FG3 and FG5 in Figure 4-6
  - **Predictability**: This is an unexpected response. This is referenced in example 2 in Figure 4-32 where an unknown condition PR4, provokes an unexpected response PR 6

- **Problem V2**: The preconditions specified one of two modes. VCR on, and VCR in timer record mode, prevent any action, except pressing the “check prog” button.
  - **Controllability** – The users actions are limited by the current mode. All other goals cannot be accomplished until the user escapes timer record mode. This is referenced in Figure 4-20
  - **Predictability** – Again the user can check the program, but they can’t do anything else, even though they can cycle through them. Corresponds again to Figure 4-32

- **Problem V3**: If all the fields contain values in whatever record is displayed, the flashing field disappears.
  - **Consistency** – This is a less obvious problem. A difference regarding whether the fields are empty or not, causes a difference in the display, but otherwise has no effect on anything else. This potential problem corresponds to Figure 4-41

- **Problem V4**: User must hold two buttons for 3 seconds before a record is deleted and when it is, no undo is possible.
  - **Directness** is violated, because of the slow continuation to the users actions. This problem corresponds to Figure 4-22.
- **Responsiveness** -- No sign is provided on the state of the action while the buttons are held. This problem corresponds to Figure 4-11.
- **Forgiveness** -- No warning is given that this is an irreversible action. Figure 4-5
- **Error Recovery** -- This cannot be undone. This problem corresponds to Figure 4-9.

**VCR Panel - Set Items - Part 1 of 2**
Set Items is assigned as a goal, because in the Panasonic SD20 manual, it is assigned as a single step covering the setting of each value. This process is too big to be modelled into a single page diagram so it is broken into two parts in diagrams, Figure 5-31 and Figure 5-32. The formatted source material for this process is as follows.

**User Goal Set Items (PART 1) -- Formatted Method**
User may [set] the **Program Position**
By choosing to [set] an auxiliary value **Note: an external source or a channel**, the user must **pick a desired value from a set of values** in each case. Whether the user [sets the values] or not to continue the user must move on to the **Date** by [pressing] the **Next Button**

If the **Date** is (empty) the **Date** is set to the [Current Date]. Regardless, the **Program Position** [stops flashing], the **Date** begins [flashing] and the set of Values [] the user can choose from a common set of (Daily, Weekly, and Date values)

User may now choose to [Set] date or [Move] to the **Start Time Hour**
To [set] the **Date**, the User must decide to [set] a **Weekly recording**, a **Daily Recording** or a normal **Date** recording. In each case the user must **pick a desired value from the current set of values**.
Whether they [set] the **DATE** or not to continue the user must [Move to] the **Start Time Hour** Values by [pressing] the NEXT button. If the **Start Time Hour** is (empty) it is set to the [current hour] Regardless, the **Date** [stops flashing], the **Start Time Hour** [starts flashing] and the possible **Values** are set to a range between [0, 23], corresponding to the hours in the day

The User may now choose to [Set] the **Start Time Hour**, or [Move on to] the **Start Time Minute**.
To Set the Start Time Hour they must **pick a desired value from the current set of values**. Either way the user must move on to the **Start Time Minute** by pressing the **Next Button**
A GSE representation for the first part of this sequence Figure 5-31, demonstrates the linear nature of the process. It is now possible to analyse this sequence in terms of potential problems.

**Attribute Analysis for Set Items Part 1**

**Problem V5:** Is a violation of Directness as in Figure 4-23. What happens is that the user, decides to enter an auxiliary channel. In the VCR these are sources, AV1, AV2, and AV3, representing external input. The alternative is to enter an ordinary channel, between 1 and 99. However both actions use the same list of values. So one or the other can’t be directly set, the user must navigate a unified list in both cases. This is made more of an issue by the called process – pick value from desired value, the mechanics of which cause a very slow progression through the data. It could be argued that by integrating the lists, the design is consistent in its approach, that may be the case but it is does not overcome the problem caused by bunching all three together.

**Problem V6:** Is the same problem as in Figure 4-23. The values from which are user sets weekly recording (Once a week), Daily recording (Once a day over a period of fixed days) and ordinary recording (a date in the month), are all picked in exactly the same way. This is exacerbated by the fact that the same list is used for three types of data.

**VCR Panel - Set Items – Part 2 of 2**

In part 1 of the Set Item method, values for the programme position (the channel), the Date and the Starting Hour were all set. In the second part of the process, values are assigned for the starting time minute, ending time hour, ending time minute, and the optional value speed. The user also has the choice to either begin timer recording, or return to set more timer programs. The process, and the associated steps may be found in Figure 5-32. This process is much the same as part 1.
The formatted version of this process is as follows:

The start time hour [stops flashing], the start time minute starts [flashing], and the list of values is set to a range of 0 to 59 minutes. The user may set the start time minute, which they do by picking a desired value from the set of values. Either way the user must press the next button. If the ending time hour is (empty) it is set to the [current hour]. The Start Time minute stops [flashing] the End Time Hour starts [flashing] and the possible values are set to [0 23], one value for each hour.

The user may then set the ending Time hour by picking a desired value from the set of values. Either way the user must then decide to [move on to] the end time minute. The user must then [press] the next button. If the End Time Minute is (empty) it is set to the [Current minute]. The End time hour [stops flashing], the end time minute starts [flashing], the values are set between [0 and 59] for each minute. The user may choose to [set] the End Time minute by picking a desired value from a list of values.

If the user is ready to stop entering values the goal is [accomplished]. However, the user may decide to move on to the speed value by [pressing] the next button. The End time minute [stops flashing], the Speed starts flashing and the values now contain only two values [SP and LP].

The user can then choose to [set] speed by picking the value from values. The user may also decide that they are finished setting these values and thus the goal is [accomplished], or they can choose to [Move to] the Program Position Value by [pressing] the Next Button. If so the Speed [stops flashing], Program Position [Starts flashing], and the Values are [set] to a combination of the Auxiliary and Channel values. The process then begins afresh by returning to the first node in Set Items Part 1.

Attribute Analysis

When an analysis of the second part of the process is conducted, outside of some issues already discussed, like Efficiency and Directness, only one significantly new problem stands out. This has a great deal to do with the supporting material, on which the process is based, which does not mention that a value can be corrected by repeating the process. But this means setting the optional value speed, and then pressing next again to return to the next value. So the main problem here is labelled problem V7.
Problem V7:
- The problem is that Start Time Minute and End Time Minute can be set to the same default value. This means, an invalid timer recording is a possibility. This is a violation of Forgiveness, since it may actually precipitate an error rather than correct it. This problem corresponds to both Figure 4-7 and Figure 4-6.

Problem V8:
- Directness: To correct any previous value, the user must move to speed and hit next again. This problem corresponds to Figure 4-25
- Responsiveness: There is no screen content to suggest that this is possible. Figure 4-12
- Predictability: Pressing next one more time, takes the user back to the first value, and not on to the NEXT record. Figure 4-32
- Learnability: The support material makes no mention of the ability to correct existing value this way.

Attribute Analysis— Pick Desired Value from Values
The final sub-goal of the methods that directly concern timer recording with the VCR Panel is “pick a value from values”. This is the closest of all the methods to its NGOMSL counterparts and contains a verify operator (an operator specific to NGOMSL) as used in the Set a Value to a Value method in Figure 5-15. The “pick a value from values” method’s GSE conversion is in Figure 5-33. When analysed against the attributes in Chapter 5, two potential problems are found.

Potential Problem V9:
The problem here is a result of problem C. The user must verify that the value is flashing so they may alter it. But is this a consistent assumption. Looking at problem C, the answer is no. When a record is first displayed, the first field is not flashing, but it is editable. This is where the model has benefited from an NGOMSL statement, perhaps owing to the fact this method is at the model’s lowest level of detail. This violates
Consistency – the Display will not always flash (see Figure 4-40).
Responsiveness - When the value isn’t flashing the user has no visual indication that the item can be changed. See Figure 4-12.

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.0</td>
<td><strong>Goal</strong> for method [set item]</td>
<td>VCR 3.1, User 3.1</td>
</tr>
<tr>
<td></td>
<td>VCR Panel displays the current value</td>
<td></td>
</tr>
<tr>
<td>3.1</td>
<td><strong>VCR Panel</strong> is showing the current item in VALUES</td>
<td></td>
</tr>
<tr>
<td>3.2</td>
<td><strong>Verify VCR Panel Shows Current Value is Flashing</strong></td>
<td></td>
</tr>
<tr>
<td>3.3</td>
<td>User decides if the current item [is less] than their desired value,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>and [presses] the + button</td>
<td></td>
</tr>
<tr>
<td>3.7</td>
<td>The <strong>Current Item</strong> on the VCR Panel [is incremented]</td>
<td></td>
</tr>
<tr>
<td>3.4</td>
<td>User decides if the current item [is less] than their desired value,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>and [presses] the - button</td>
<td></td>
</tr>
<tr>
<td>3.9</td>
<td>The <strong>Current Item</strong> on the VCR Panel [is incremented]</td>
<td></td>
</tr>
<tr>
<td>3.5</td>
<td>User decides if the current item [is less] than their desired value,</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Goal</strong> is [accomplished]</td>
<td></td>
</tr>
</tbody>
</table>

**Assumption** – That the Values ++ and Values – are automatically looped to the first and last values when they reach the end and start of the values list.

Figure 5-33: Timer Recording: Pick a Value from Values

**Problem V10:** In Problem J The user must decide whether to move forward or back in the list, with no feedback on what comes before or after, other than their own knowledge. This violates

- **Responsiveness** – The user sees only one item at a time. This is not a big problem with numeric values, but speed, channels, and date all contain non-numeric values.
- **Predictability** – Will users get the value they are looking for? See Figure 4-33
- **Directness** – Values cannot be chosen directly This problem corresponds to example Figure 4-26.
- **Efficiency** – In long lists this selection method is likely to take time. Corresponds to problem in Figure 4-29

This concludes the VCR Panel only methods. However this does not end the modelling of this device since the tasks can also be achieved by using both the bar code sheets and the remote.
5.6.3 Using the Remote Control

The second method for programming timer programmes is by using the remote control. Though it follows a similar pattern in terms of data entry, the first major difference is that it employs an additional display device; an LCD Screen on the remote control. The second major difference, is while checking values is much the same as using the VCR panel, setting values involves transmitting the data to the VCR, only after they have all been entered and viewed on the remote’s display device. A third difference is that remote timer programming (using the buttons only) has to share its LCD with the other timer recording process (using the bar code scanner) since the methods are not interchangeable.

Checking Records with the Remote

To check a recording using the Remote the user must check that the display is not on. If it is off, they can [press] the transmit button. If it is on they must turn it off. They do this by pressing either the scanner/on off button or the PROG button. They can then also press the transmit button.

Figure 5-34 contains a reworking of the VCR panel Timer recording model to accommodate the checking record method alongside the original checking methods using only the VCR panel. This based on descriptions on page 27 of the Panasonic SD20 manual. If the user chooses to use the remote as a means of selecting a channel, they have to make sure that the display is off. In an additional step the “VCR Check clock status” event is added, to ensure continuity between the nodes.

Attribute Analysis

A quick examination of the new method highlights some additional complexity that was not present when using the VCR panel only. These two problems are classified K and L since the same actual VCR system (the SD20) is being analysed, and are in addition to those problems already discussed.
Problem R1:

There are several problems unique to the remote control. First of all, the user must check that the display on the remote control is off, before pressing the transmit data. This adds an extra step to the process. The consequence of not checking, it and pressing transmit, is that an incorrect code will be sent.

This impacts on

- **Directness**, as an extra step is added to the process Figure 4-26.
- **Forgiveness**, since no warning is given
- **Predictability**, because pressing transmit without checking – will bring unforeseen consequences, depending on whatever data or instruction is on screen.
- **Error Recovery**. This function is irreversible.
Problem R2:
There are two buttons that turn off the Scanner. One is PROG - the programme
Button, and the other is SCAN-On Off. Both have additional functions associated
with them.

- **Simplicity** - The process is made more complicated by multiple system states
  Figure 4-36
- **Familiarity** - This is significantly different to the original VCR panel process.
  Figure 4-39.

Translation: Setting the Program Position on the remote control. Figure 5-35
The **user** must [press] the **Prog Button** to turn the LCD display on. If the LCD display is
(on) this will turn the LCD display [off] and it the **user** will have to [press] the **Prog Button**
again. The **LCD display** is set to [timer programme mode] It contains a **Prog-Position** which
starts with a value of 1 and begins [flashing], **Starting time hour**, **Starting time minute**,
**Ending time hour** and **Ending time minute**. To [set] a Timer Programme, the **user** must
**enter the programme position**, **enter the date**, **enter the start time hour**, **Enter the start time
minute**, **enter the End time hour**, and **enter the end time minute**.

After setting these values the **user** must [press] the **transmit button**. The **LCD display**, then
passes each of these values (prog, start time etc) to the **VCR panel**, and the **VCR enters**
[timer record mode]

**Problem Analysis**

**Problem R3**: is both a Simplicity and a Directness issue. The instructions specify that
the display must be off, turn it on.

**Problem R4**: is a Consistency and Controllability issue. Unlike the VCR method, the
LCD display does not contain a Timer Programme Index. The VCR user cannot set
this on the LCD display in the same way it appears on the VCR Panel.

**Problem R5**: is a very clear Controllability and Predictability issue. Controllability is
violated because the system automatically goes into timer record mode without
the user having intended it to. Predictability is violated because this is an unexpected result, considering the user's aim had been to transmit values to the VCR. The other attribute affected is Directness, given the user is unable to set other timer programmes without cancelling the timer recording.

Sub Goal: Entering a Program Position – Translation (in Figure 5-36)

The User may choose to set a program position or move on to set the LCD date instead. If they choose to set the program position they can choose between setting an (auxiliary channel) by [pressing] the Input Select Button, which sets the Program Position Value to A1. On the other hand if the user wants choose a (standard channel), they must decide whether to enter a One Digit Channel, or a Two Digit Channel. To enter a one digit channel, the user must [press] the +/- Button, Enter the first digit, of the Program Position, before entering the second digit of the Program Position. To enter a single digit however they enter a single digit, the first digit of Program Position is set to 0. In both cases once the digits are entered the user can press move on to the LCD date value by [pressing] the Next Button. Once the Next Button is pressed, the LCD Programme position is [not flashing], and the LCD Date starts [flashing]. The LCD Program Position is now set.
**Problem Analysis**

**Problem R6** is a Directness, Simplicity and Efficiency problem. The strange way of entering two-digit and one digit numbers is unusually complex (for a simple numeric value), indirect for standard numeric entry, and inefficient due to its additional steps. It is also lacks Consistency with other button sequences and Familiarity to methods in both the NGOMSL model (set value to value) or any method on the other VCR.

### 5.6.4 Using the Bar Code Reader

The third way of timer recording, adds some additional complexity to the timer recording tasks. Like the remote control, the LCD display must be on. However there are two LCD modes (scanner on) and (scanner off).

Goal Set Items with Barcodes: The user [presses] the **scanner on off button**, and if the LCD Display is (off), the **scanner** will be [enabled] and the **LCD display** will come on in [Scanner Mode]. If the display is on, this will turn the **display (on)**, and the **scanner (off)**, and the user must [press] the **scanner on off button** again. Once the LCD Display is on the user may now **Enter a timer programme**, **Check a timer programme**, or cancel (delete the current program). If the user decides to check a timer programme, they must **trace the check bar code**. The LCD Display displays a [C] for cancel. The user can press transmit. If the **VCR panel** is already in (timer programme mode), the **VCR Panel** displays
the next item in the Timer Programmes collection. If the VCR panel is (not in timer programme mode), it will be put into timer record mode and display the first item in the Timer Programmes collection.

If the user decides to cancel a bar code, the user must trace a cancel code and press the transmit button. If the VCR panel is in [timer programmes] mode. The [current] timer programme will be [cleared]. The VCR will [beep], and the VCR panel will display the [current] timer programme.

Figure 5-37: Setting the remote to Scanner Mode
PROBLEM ANALYSIS

Problem B1: This is a Consistency (Figure 4-40) and Responsiveness (Figure 4-15) issue. If the LCD display is on, the scanner is not necessarily on. But if the scanner is on, the display must be on. But there is no way of telling whether the scanner is on or not.

Problem B2: There are two factors here. The first is Predictability (Figure 4-31). The button will act differently depending on the mode the VCR is in. The second is Controllability (Figure 4-19), as timer record mode is activated, regardless of the users present goal.

Problem B3: is an Error Recovery (Figure 4-9 issue. Cancellations cannot be undone.

If the Scanner is on, codes can be swiped. Instead of using the up and down buttons on the VCR panel, or buttons on the remote, the user must trace the scanner across codes on the accompanying bar codes sheet. No next button need be pressed, because the scanner does so automatically. So any next button presses are negated.

Translation: Entering the Timer Programme values [Figure 5-39]

The user must first [enter] the programme position. The user must decide whether to [set] an external signal, by tracing an AV Code, and [setting] the LCDs Programme position to the [AV code] or a [normal channel] by tracing a Channel Code, and [setting] the LCDs Programme position to the [Channel Code]. The user must next [enter] the date. The user must [decide to use] a weekly code, by tracing an "everyweek" code and setting the LCD DATE to an [every week date] or the user can [decide to use] an [everyday code], by tracing an everyday code, setting the LCD Date to an [everyday code], or the user can [decide to use] an ordinary date, and trace a date value setting the LCD date to a [standard Date]. The user must then [set] Start Time, by tracing a start time hour, and [setting] the LCD start time to the [start time hour] then tracing a Start time minute code, and setting the LCD Start Time minute to the [trace value]
Translation: Tracing Values

[Figure 5-38]

User must find the code (on the barcode sheet). The user must then [swipe the code]. If the scanner (accepts the code), the remote beeps. If (not) nothing happens and the user must [swipe] the code again. If the code is [accepted], the scanner [interprets] the code and outputs this to the current item on the LCD display.

Problem Analysis
Problem B4: This is a Responsiveness, Learnability and Error Recovery issue since no feedback is given if the code not swiped properly.

Problem B5: To enter any more programmes, the codes need to be wiped by swiping the cancel code. This classifies as a Directness, Efficiency, Consistency and Familiarity problem. This is because this is an indirect and ultimately inefficient means of entering values. This process is not consistent with the other methods on this device, and unfamiliar in relation to the other timer programming techniques and the NGOMSL model in 5.4.
This concludes the translation of methods and identification of usability problems, using GSE notation, for the HD-SD20 VCR. It is now possible to collate the findings.
5.6.5 Summary of Problems identified on the SD20

There are 21 potential usability problems identified across the GSE models using the usability attributes modelled in Chapter 4. These problems correspond to the various tasks in the user study using the Bar Code reader on the remote control. Table 5-1 contains these problems, and the usability attributes pertinent to the problem. These attributes in the top row are identified by codes, which are fully defined in both Table 5-9 and Appendix B.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Attributes</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1</td>
<td>RS ER FG PR</td>
<td>Clock</td>
</tr>
<tr>
<td>V2</td>
<td>CT PR</td>
<td>Timer Record</td>
</tr>
<tr>
<td>V3</td>
<td>CS</td>
<td>Flashing Field</td>
</tr>
<tr>
<td>V4</td>
<td>DR RS FG ER</td>
<td>Two button delete</td>
</tr>
<tr>
<td>V5</td>
<td>DR</td>
<td>Channels</td>
</tr>
<tr>
<td>V6</td>
<td>DR</td>
<td>Dates</td>
</tr>
<tr>
<td>V7</td>
<td>FG</td>
<td>Delete to correct</td>
</tr>
<tr>
<td>V8</td>
<td>DR RS PR LN</td>
<td>Start time/end time</td>
</tr>
<tr>
<td>V9</td>
<td>CS RS</td>
<td>Flashing Field</td>
</tr>
<tr>
<td>V10</td>
<td>RS DR PR EF</td>
<td>Previous and Next</td>
</tr>
<tr>
<td>R1</td>
<td>DR FG PR ER</td>
<td>Display on off</td>
</tr>
<tr>
<td>R2</td>
<td>SM FM</td>
<td>Scan – Prog</td>
</tr>
<tr>
<td>R3</td>
<td>SM DR</td>
<td>Display off</td>
</tr>
<tr>
<td>R4</td>
<td>CS CT</td>
<td>No index on LCD</td>
</tr>
<tr>
<td>R5</td>
<td>CT PR</td>
<td>Automatic Timer Record Mode</td>
</tr>
<tr>
<td>R6</td>
<td>DR SM EF CS FM</td>
<td>Program position 2 digits</td>
</tr>
<tr>
<td>B1</td>
<td>CS RS</td>
<td>LCD / Scanner</td>
</tr>
<tr>
<td>B2</td>
<td>PR CT</td>
<td>Automatic Timer Record and different button modes</td>
</tr>
<tr>
<td>B3</td>
<td>ER</td>
<td>Can’t fix cancellation</td>
</tr>
<tr>
<td>B4</td>
<td>RS ER LN</td>
<td>Tracing error</td>
</tr>
<tr>
<td>B5</td>
<td>DR EF CS FM</td>
<td>Wipe codes for more programming</td>
</tr>
</tbody>
</table>

Table 5-1: Usability Problems found across the HD-SD20 Task Models.

V – VCR Panel; R Remote Control; B- Bar Code Scanner.

To determine the success of the attributes, and GSE models, as a means of predicting usability problems, these problems will be compared to the findings of the user tests later in the study. However, the problems on the second VCR device must also be identified. These relate to tasks 1 and 2 in the user study, which cover alternate methods for timer programming using an On-Screen-Display and G-Codes.
5.7 The Panasonic NV-HD20mk2a VCR model

The HD620, a more recent VCR model, is significantly different to the SD20. Instead of presenting all the information on the VCR or on an LCD screen, the HD620 uses an onscreen display (or OSD) as it is referred to in the manual, which appears on the television screen. In order to timer record a program, two options are presented. The first is to input the data via the remote. The second is to use G-Codes. G-Codes are a sequence of numbers, which carry all the information needed to timer record. G-Codes can be found in most television guides, including the one used in the usability study. G-Codes are entered by using the buttons on the remote control. The buttons that correspond to G-Code and Timer Recording are shown in Figure 5.40.

![Diagram of the NV-HD620mk2a remote control button layout.]

The first task, Timer Recording (without the G-Codes) uses the same information as that needed by the SD20. The users must enter a channel, a data, a starting timer (On) and an ending-time (Off). Changing the speed (SP or LP) as part of this process is optional. The programme position is provided automatically, much like the SD20. However, this can only be changed on a second screen, used for checking and reviewing how many timer recordings are listed. The HD620 VCR supports up to 8 different timer programmes, each with a different index or “programme” number.

Both methods make use of the On Screen Display, which is able to be accessed using menu keys on the remote. Some keys are available via a side panel on the TV, but since this study did not test any method using these buttons, Tasks, 1 and 2 relied on the remote control entirely.
5.7.1 Using the On Screen Display

The OSD (the On Screen Display) allows the user to set the values on the television set, allowing for additional content such as the button images in Figure 5-1. A number of pre-requisites are specified in the Panasonic HD20 manual and which are covered in following translation for Accessing the Timer Recording Screen and its GSE representation in Figure 5-42.

Translation - Accessing Timer Recording with the OSD

The following and all subsequent descriptions of the OSD procedures are based on pages 24 and 25 of the Panasonic HD20 manual.

The TV must be (On). The TV Viewing Channel must be (Selected). The VCR must be (On) and the Clock must be (set). The user must [press] the "Check Prog" Button. The OSD appears in [Program Edit Mode] containing values for Prog Position, Date, Start Time, End Time and Speed. User may now either enter timer recording values or check another program.

Attribute Analysis.

There are no obvious errors with this small excerpt of the OSD behaviour. The user may choose now whether to enter a timer recording or to check another timer recording value.

Translation for Entering a Timer Recording

Part I

The formatted description of the procedures for Entering Timer Recording Values using the OSD is as follows:

They may choose to set the Program position. But if (nothing is pressed for two seconds), or the user [presses] the date up or date down keys), the date item will begin [flashing]. The user must now [decide] if the Program Position (is less than their
desired value), and if so the user must [press] the CH UP Button which changes the Program Position value to the [next] Program position value, or the user may [decide that the value is higher] and [press] the CH DOWN button setting the Program Position to the [preceding] value. Otherwise, if they think the Program position value is (correct) they can wait, for the Date field to begin [flashing].

Problem Analysis – Part 1

Problem 01: is the same problem as that encountered on the other VCR. The setting of the clock will cause problems down the track if it is not set.

Transcription Part 2

User must now [set] the Date, or [move] on to the next field. The user must decide whether they want to [enter] a weekly recording date, a daily recording date or a normal date. They must [check] the current date and decide (if it is less than the current value) and press Date Up, (more than the current value) then Press Date Down, or if it is (equal to it) in which case they move on to the Start Time.

Figure 5-43: Entering a Timer Recording Part 1
Problem Analysis

There are several possible problems with the process

Problem O2: The same procedure for both Auxiliary and Standard Channels may prove a Directness problem.

Problem O3: Controllability is an issue if the field automatically advances

Problem O4: The same procedure for each type of date may prove a Directness and Efficiency problem.

Problem O5: Predictability is an issue here. How does the user know which value proceeds or precedes the value displayed?

Problem O6: Same as O5

Translation for Entering a Timer Recording Part 2

The user must now decide if they want to [enter] a Start Time, or [move on] to the next field. To set the start time, they must decide whether their desired value is (higher) or (lower) than the start time displayed. If lower the user must either [press] the CH UP Button which increments the Start Time by [1 minute] or [hold it down] incrementing the Start Time by 30 minutes until they [release] the button. If the value is lower the user must either [press] the CH DOWN Button which [decrements] the Start Time by [1 minute] or [hold it down] [decrementing] it by [30 minutes] until the user [releases] the button when they are [finished or time elapses with no activity], they must [move] onto the Ending Hour. The user may then [decide] whether to [enter] another timer program by [pressing] the “Check Prtg” button again (returning them to the start of Timer Recording part 1), or [activate] Timer recording by pressing the “Timer Rec” Button which sends the VCR into [timer record mode]

Problem Analysis

Problem O7: Automatic Advancement

The only problem here is the same as B in the previous diagram, in the sense that the next Image will start flashing, if there is no activity for two seconds. This is a Controllability and Predictability problem.

Problem O8: Setting the start time with the OFF button

The button for setting the start time is called the OFF Button. This affects Predictability, and Simplicity.

Problem O9: Setting the start time with the ON button

The button for setting the end time is called the ON button. This affects Predictability and Simplicity.
Problems O10 and O11: The same as O8 and O9 but for Entering the End Time instead.

Translation for Entering or Deleting a Timer Programme and selecting next record.

The current currently selected value is set to 1. After the user [presses] the CHECK PROG button, the OSD [Timer Programmes] updates by displaying the Timer Programme List, which currently selected is set to 1. The OSD now displays Timer Programmes [1] through to [8].

The user has to option to [delete] the currently selected record by [pressing] the Cancel Button which then [deletes] the currently selected record in Timer Programmes If (there are no more records) this goal is [accomplished]. If (there are not) the goal is [accomplished]. If there are more the user must [select] the next record, by pressing the Timer Record button. The current timer record is incremented [+1] and the screen is updated again.

Figure 5-44: OSD Timer Recording Part 2
Problem Analysis

The main problem is a Forgiveness and Error Recovery issue.

Problem O12: There is no option for correcting a record, it must be deleted then re-entered. This is a Directness, and Efficiency problem.

Problem O13: There is no undo option for a deleted record. This is an Error Recovery issue.

That concludes the analysis of the on screen display tasks. The last method of timer programming is to use G-Codes. Much of this functionality is shared with the OSD tasks. However, the actual G-code system is unique.

5.7.2 Using G-CODE

G-Code programming is perhaps the most efficient means of timer programming. The user must obtain the correct code from the TV-guide (or other source), key it in, and then activate the recording. The code has time, date, and duration information that may be gleaned from the code. Figure 5-45 contains the screen where code is entered. The procedures for doing this are as follows:

Translation for Timer Recording with G-Codes Part 1 - Figure 5-46

The TV must be [On], The TV Viewing Channel must be [Selected]. The VCR must be [ON] the Clock must be Set. IF the VCR TV Switch is not set to [VCR], the user must slide the VCR TV Switch to VCR, When the VCR/TV switch is set to [VCR], the G-CODE button and numeric buttons are [enabled]. The user can now [press] the G-Code Button, which puts the OSD into the [G-CODE] screen mode.
Problem Analysis
In the first part of setting G-CODE Timer Programmes, two problems have been noted.

Problem G1: The first is the set-clock method, which is set as a pre-condition. To keep the model consistent with the methods on the other device, the same problems that will occur if the clock is not set, apply here also.

Problem G2: The second problem is unique to this process. If the VCR isn’t set to VCR-TV, the G-Code and Numeric buttons are locked. This is a Directness, Efficiency, Controllability and Simplicity problem.

Translation for Timer Recording with G-Codes Part 2- Figure 5-47
From the OSD [G-code] Screen in Part 1. The user may check a timer program this is exactly the same as the procedure carried out for the OSD method, and is therefore not repeated. The user can now enter a Timer Programme. If the user wants to correct the code, the user can [press] the cancel or rewind button to [clear] the G-code and then enter it again. If the user decides the G-Code is finished they can [press] the Enter Button. The VCR interprets the G-Code, and if it rejects it, sends a message to the OSD that the G-Code is incomplete and returns the user to the OSD [G-Code entry] screen. If the validates the code, it generates a timer programme from the g-code, containing program position.
date, start time, and end time and outputs these to the OSD. If the VCR's Channel codes are not set, the program position is set to [empty], before generating the Timer Programme. Once all the data from the timer programme has been outputted to the OSD, the user may set more programs, by pressing the G-Code button, and having the list of Timer Programmes move to its [next item] and repeat the entry process. If the user wants to correct the code, they can do this the same way they did for the OSD data. The user may also choose to put the VCR in to [timer record] mode by pressing the Timer Record button. However if the programme is (invalid), the user will have to enter the code again by returning to the OSD.
Problem Analysis.

Problem G3: The first major problem is a Responsiveness one. If the VCR rejects the G-Code it has a single message “Programme incomplete” but won’t say why. This is also a Forgiveness issue, and a Predictability issue as well.

The second problem occurs if the G-Code channel settings are not set on the VCR, as this means that the user will not be getting the full information from the G-Codes and must specify it themselves.

Problem G4: Is a Consistency, Simplicity and Forgiveness issue. The functionality of the G-Code entry system depends on G-Codes being assigned to channels. If these are not set, the program position will not be pre-determined.

Problem G5: Finally when the user activates timer record mode, and the record is incomplete, the VCR will not activate. This is a Forgiveness and Predictability issue.

Because Cancelling a Timer Record with G-CODE programming is also the same as the methods for OSD, it inherits the same problems and so will not be remodelled.

5.7.3 Summary of Problems identified on the NV-HD620mk2a

To determine the success of the attributes, and GSE models, as a means of predicting usability problems, these problems will be compared to the findings of the user tests.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Attribute Codes</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>O1</td>
<td>RS ER FG PR</td>
<td>Clock</td>
</tr>
<tr>
<td>O2</td>
<td>DR</td>
<td>Auxiliary</td>
</tr>
<tr>
<td>O3</td>
<td>CT</td>
<td>Automatic field advance</td>
</tr>
<tr>
<td>O4</td>
<td>DR EF</td>
<td>Dates</td>
</tr>
<tr>
<td>O5</td>
<td>PR</td>
<td>Order of values</td>
</tr>
<tr>
<td>O6</td>
<td>PR</td>
<td>Order of values</td>
</tr>
<tr>
<td>O7</td>
<td>CT PR</td>
<td>Automatic Advancement</td>
</tr>
<tr>
<td>O8</td>
<td>PR SM</td>
<td>Start Time On</td>
</tr>
<tr>
<td>O9</td>
<td>PR SM</td>
<td>Start Time Off</td>
</tr>
<tr>
<td>O10</td>
<td>PR SM</td>
<td>End Time On</td>
</tr>
<tr>
<td>O11</td>
<td>PR SM</td>
<td>End Time Off</td>
</tr>
<tr>
<td>O12</td>
<td>DR EF</td>
<td>Cancel to Correct</td>
</tr>
<tr>
<td>O13</td>
<td>ER</td>
<td>Can’t Undo</td>
</tr>
<tr>
<td>G1</td>
<td>RS ER FG PR</td>
<td>Clock</td>
</tr>
<tr>
<td>G2</td>
<td>DR EF CT SM</td>
<td>G-Codes locked VCR-TV Button</td>
</tr>
<tr>
<td>G3</td>
<td>FG PR RS</td>
<td>Programme incomplete errpr</td>
</tr>
<tr>
<td>G4</td>
<td>CS FG SM</td>
<td>G-code assignment to channels</td>
</tr>
<tr>
<td>G5</td>
<td>FG PR</td>
<td>Timer Record mode, incomplete record</td>
</tr>
</tbody>
</table>

Table 5-2: Problems found on NV-HD620mk2a VCR using GSE
later in the study. However, the problems on the second VCR device must also be identified. These relate to tasks 1 and 2 in the user study, which cover alternate methods for timer programming using an On-Screen-Display and G-Codes.

This now ends the analysis of both VCR devices. In the next part of this study, the user test results must be analysed to see what usability problems can be found with actual users using the devices that have just been analysed without users.

5.8 Pilot Study User Test

This section documents the pilot study user test, presents the results and compares it against the GSE problems identified in the previous section. Before outlining these findings, it is important to specify the conditions and processes that were used to obtain them.

Participants.

A survey (contained in Appendix G) was handed out to 30 people, including students at the science, technology and art faculties at Griffith University, local high school students, staff at an accountancy practice, and friends and family of those already mentioned. Six subjects were chosen based on criteria specified in 4.5.1. The experiment took approximately 1.5 hours to complete. Each participant was paid $20 AUD for their time.

Apparatus and Setting

The setting and room set-up for the user tests, as well as the apparatus used are discussed in detail in sections 4.5.4 and 4.6.1 of Chapter 4: Methodology. These include the two video cameras, phone devices and manuals.

Tasks

The task sheets including instructions for the study may be found at the end of Appendix B. These tasks corresponded to the five different ways of timer recording. Each task involved specifying several types of timer recording, including different instalments of a program over multiple days, correcting one of these programs, and then deleting all of them from memory. A short summary of these tasks is as follows:
Tasks on the Video Recorder: NV-HD620

Task 1: Using the On Screen Display (OSD)
Subjects used the menus displayed on the television set to specify types of
timer recordings (one day, multiple days, and once a week). They also had to
extend the duration of the first program, before deleting them all.

Task 2: G-Code Programming
Subjects entered G-Codes, pre-set numeric values containing timer-recording
settings for specific television programs. These were supplied in a television
guide. Subjects had to enter two G-Codes, and then find a way to enter a
program on consecutive days. Finally they had to delete the programs from
memory.

Tasks on the Video Recorder NV-SD20

Task 1: Using the VCR panel.
Subjects used a panel on the front of the VCR, consisting of a digital display
and several buttons to enter, correct and delete programs much like those
specified in the On Screen Display task.

Task 2: Using the Remote Controller.
Subjects used the remote control, its inbuilt display panel and transmission
button, to perform the same tasks as before but for different programs.

Task 3: Using the Bar Codes.
Subjects used the remote controls in built bar-code scanner, as well as a bar-
code sheet containing codes for various VCR actions (such as correct and
delete) and numeric values (to specify times). The tasks performed were
similar to the previous task, but for different programs.

Test Procedure
Six subjects performed the five tasks on both VCRs in individual sessions conducted
over a two-day period. Subjects were instructed to express their thoughts aloud, as
they conducted the task. They were also informed they were able to ask questions of
the researcher relating to clarification of an instruction or any non-task related
problems but additional help could not be provided to complete a task. The task
sheets (at the end of Appendix B) and instructions were read out at the start of each of
the VCR tasks. Subjects were asked if they had any pertinent questions about what they had to do. The sessions then commenced.

The test sessions were recorded on a single VCR camera 15 minutes were assigned to each task, with an additional 5 – 10 minutes if necessary. Students were verbally told when the 15-minute mark was reached. They were then told to move on 10 minutes later, if the subjects chose to use the extra time. 2 hours was allocated to perform all the tasks, including the extra time and time to read the instructions.

When the time was up or the students were finished they were instructed to complete the post study questionnaire found at the end of their survey forms in Appendix G. The two videotapes were then tagged and stored for later transcription. The video tapes were then transcribed on to talk aloud protocols, according to procedures specified in 4.7.3.

**Problem Identification and classification.**

The talk aloud protocols were passed on to the examiners who were instructed to identify potential problems in the protocols, mark them, comment if necessary, and then record both the attributes they believed related to the problem and the severity of the problem. The examiners were supplied with an information pack (see Appendix B) containing instructions and information for marking potential problems.

Following these instructions, the examiners marked sections of the talk aloud protocols (supplied in Rich Text format) that they believed constituted a potential usability problem. Each protocol corresponded to talk aloud results for one of the subjects in the study. Using the Insert comment feature of the reviewing toolbar in Microsoft Word, the examiners marked the text with a severity rating between 1 (a critical problem) and 4 (a very minor problem). They also added codes corresponding to the attributes they believed applied to the problem. The descriptions and codes for these attributes may also be found in Appendix B. Thus if an examiner believed a particular problem was manifested in one part of the talk aloud protocols, they marked the area, assigned one or more attribute code and a severity. So if examiner D, believed a potential problem (such as subject Z not being able to exit a dialog) violated the Directness and Controllability attributes and was serious enough to
prevent the completion of a task making it a critical problem, they would mark the area with 1DRCT. This translates to 1 for critical severity, DR for Directness and CT for controllability.

Collating the Potential Instances of a Problem

Once the marked protocols were returned to the researcher, the comments for each examiner were numbered for statistical purposes. Table 5-3 contains comments 42 – 50 for Examiner A. The first row shows that the 42nd potential problem marked by Examiner A, was rated severity 1 and assigned the attributes Responsiveness (RS) and controllability (CT).

<table>
<thead>
<tr>
<th>Prob</th>
<th>Severity</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>42</td>
<td>1</td>
<td>RS CT</td>
</tr>
<tr>
<td>43</td>
<td>3</td>
<td>FG EF ER</td>
</tr>
<tr>
<td>44</td>
<td>3</td>
<td>RS US</td>
</tr>
<tr>
<td>45</td>
<td>3</td>
<td>PR</td>
</tr>
<tr>
<td>46</td>
<td>3</td>
<td>EF</td>
</tr>
<tr>
<td>47</td>
<td>2</td>
<td>PR</td>
</tr>
<tr>
<td>48</td>
<td>2</td>
<td>RS</td>
</tr>
<tr>
<td>49</td>
<td>3</td>
<td>FL</td>
</tr>
<tr>
<td>50</td>
<td>2</td>
<td>RS ER</td>
</tr>
</tbody>
</table>

Table 5-3: Comments for Examiner A

The protocols were then collated. At this point of the process three versions of the same protocols existed for each subject, one for each of the examiners. These were collated into a single document for each user test session, so it could be determined which potential problems overlapped each other, and which stood alone. These were then compiled into problem ‘instances’, meaning areas of a session where potential problems were identified. If the different examiner comments overlapped or intersected on the talk aloud protocols, they became a single instance of a problem. To easily classify which examiners comments applied, each instance contained the examiners initial, and the number of the comment. Brackets were used to mark overlapping comments. For instance D48[A42] indicates that the section of the protocols marked Examiner D comment 1 also contains a subsection marked as Examiner A’s comment 42.

Once all these instances had been collated, these were grouped together by the task in which they were found. Thus potential problems identified by all three examiners for all six subjects were now grouped together and assigned into each user test task. This meant five lists of problem instances now existed for the G-Code, Bar Code, VCR, Remote, and OSD tasks respectively. These instances were then expanded to reveal all the severities and attributes for a given instance of a problem. This yields a table.
like Table 5-4. Each row of this table contains an ID number for the instance, the subject code for which this instance was recorded, the examiner comments that make up this instance (examiner code and number), the severity and attribute codes contained in these comments, and an average severity (the sum of all the severity values divided by the number of severities in a problem) across all examiner comments. These averages were used to order the rows in the task tables.

<table>
<thead>
<tr>
<th>Instances</th>
<th>Subjects</th>
<th>Comments</th>
<th>Severeities and Attributes</th>
<th>Severity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>O</td>
<td>D48[A42]</td>
<td>1DRTN[1RSCT]</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>R</td>
<td>D64[A64]</td>
<td>2DRUS[1SMFG]</td>
<td>1.5</td>
</tr>
<tr>
<td>3</td>
<td>Z</td>
<td>D19</td>
<td>2ERTC</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>O</td>
<td>A41 S56</td>
<td>2CT 2CTPR</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>T</td>
<td>S62 A50</td>
<td>2RS?SM? 2RSER</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 5-4: Problems Instances 1 – 5 for G-Code Task

Extracting a Set of Common Problems

Now that a list of possible problems existed for each task, it was important to classify these into a set of concrete problems for the task. After all, many of these instances, would be describing the same problem, but for different subjects. The instances in tables like Table 5-4 were compared against the protocols, task sheets and videotapes to see if they were describing the same problem. One problem could correspond to many of the instances. For instance, the first problem in the G-Code sequence, where the user enters an out of date G-Code, and is only told the G-Code is incorrect, corresponds to instances 1, 2, 4, 5, 8 and 15 from the G-Code talk aloud protocols.

<table>
<thead>
<tr>
<th>#</th>
<th>Problem Description</th>
<th>All Codes</th>
<th>S</th>
<th>Ex</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><strong>Incorrect Code [1,2,4,5,8,15]</strong></td>
<td>1DRTN[1RSCT] + 2DRUS[1SMFG] + 2CT 2CTPR + 2RSSM 2RSER + 3ERFGRS 2PR + 3LNER[3CT]</td>
<td>RO TI OO</td>
<td>DA DA DA AS SA ASDS</td>
</tr>
<tr>
<td></td>
<td>The VCR responds to an out of date code</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>with “Code Incomplete.” Users would then</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>type it again thinking it was their mistake.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td><strong>Exiting Timer Record Mode to Cancel</strong></td>
<td>4DR + 4 LN</td>
<td>ZZ</td>
<td>DD</td>
</tr>
<tr>
<td></td>
<td>[21,22]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Timer Record mode, locked down the VCR.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>There was only one way to stop it,</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>and until then all other buttons were</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>suspended. The subjects often found it</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>difficult to exit.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5-5: Compiled Problems 1 and 2 for G-Code Task
Thus a list of problems was compiled for each task listing which instances these problems corresponded. The example in Table 5-5 contains a numeric identifier for the problem, a description of the problem (containing a title, description, and all the instances that apply), the corresponding severity and attribute codes for each of these instances added together, the corresponding subjects, and corresponding examiners. Subject codes and Examiner codes are repeated, since multiple instances for one subject or one examiner could be part of the same problem. Plus (+) symbols are used to divide the different instances in the All Code column.

The first row of Table 5-5 shows that this problem corresponded to entering an out of date code when using the G-Codes. According to the description of the first problem “Incorrect Code”, it consisted of problem instances, 1,2,4,5,8,15. Thus the first three code sequences (divided by + signs) in the “all codes” column correspond to instances 1,2 and 4 in Table 5-4. Problem 2 in the G-Code table only corresponded to two problem instances 21 and 22 for the same subject Z and the same examiner D. Looking at the all codes column this problem has an average severity of 4, and examiner D believed it applied to Directness (DR) and Learnability (LN). These last two columns tell us that problem 1 was found in the protocols corresponding to three subjects (R, O and T) and three examiners (D, A and S) whereas problem 2 applied to one subject (Z) and was reported by one examiner (D). Because several incidents may constitute a problem, the last two columns of the row for problem 2 also tell us that the two incidents involving subject Z (hence the two Zs) were reported by examiner D (hence the two Ds).

**Graphing the Attributes for a Problem.**

It was now possible to graph the severity and attribute use for a given problem, visually demonstrating the prevalence of a particular attribute. Graphs were constructed for each of the problems, to better illustrate which attributes the examiners believed applied. In Figure 5-48 the frequency and severity of attributes for

![Figure 5-48: Figure G-Code problem – Incorrect Code](image-url)
the first G-Code problem are shown. The Y (vertical) axis to the left of the graph counts the frequency a particular attribute is cited. The X (horizontal) axis at the bottom of the grid contains the 16 two letter codes for all the attributes the examiner could mark on to the transcripts (The codes are contained in Appendix B).

The yellow boxes mark each time an attribute was cited in the instances that make up this problem. Since Controllability was cited four times four boxes are stacked vertically on top of the (CT) code that corresponds to controllability. Since Learnability is only cited once one box sits in the (LN) column. The numbers in the boxes display the severity that corresponded to the original comment in which the attribute was cited. For example, instances 1 and 2 in the all codes column for problem 1 are 1DRTN [1RSCT] and 2DRUS[1SMFG]. The square brackets, which were used earlier to indicate how the comments related to each other, are ignored. In the first instance, a severity of 1 precedes Directness (DR) and Transparency (TN). Thus the first box in the DR and TN columns is marked severity 1. The second comment in instance 1 (contained in square brackets) contains a severity of 1 plus the attributes Responsiveness and Controllability, so a box containing a 1 is added to the Responsiveness and Controllability columns. In the second instance (2DRUS[1SMFG]) for problem 1, a severity of 2 precedes Directness and User Satisfaction (US). So a second box is added to the Directness column this time with a severity of 2, while a severity 2 box is added to the User Satisfaction column. The next comment in the second instance, adds a severity 1 box to the Simplicity (SM) and forgiveness (FG) column.

This provides a visual representation of how each attribute was assigned to a problem. By looking at the finished grid it is possible to see that Controllability and Responsiveness were cited four times, once with a severity of 1, twice with a severity of 2 and another time with a severity of 3. Making them the most frequently cited attributes.

In the second G-Code problem, “Exiting timer record” two instances are recorded, containing a single instance each. In this case the first instance gives us a Directness problem of severity 4. The second instance gives us a Learnability problem of severity 4. The bar charts for all the problems in the user tests can be found in Appendix I.
Comparing the results to the GSE models

Now that the problems have been classified according to task, it is possible to compare the problems found in the user tests, with those found in the GSE models. Since the GSE models were classified according to task, the problems assigned to the G-Code models in section 5.7.2 labelled G1 through to G5, can be compared against the problems identified in the user tests.

In order to make a useful comparison, it is important to determine whether the attributes identified in the GSE models correspond to those cited in the user tests. Because examiners were able to mark multiple attributes in a single comment, and problems in the user tests could correspond to multiple comments, comparing attributes isn’t a simple 1 to 1 proposition. Thus when the comparisons were made, special attention was paid to which attributes were most frequently associated with a problem, and which were assigned the highest severity for that problem.

The user test problem in Figure 5-48 when analysed, seems to correspond with problem G3 in the GSE analysis. In the GSE analysis problem G3 was listed as a Responsiveness, Forgiveness and Predictability error.

The first value to calculate is whether the attributes in G3 were found in the corresponding user test problem. Since all three attributes RS, FG and PR were found somewhere in the user tests, we get a presence value of 100%. However, a lot of attributes were often identified in the user test results for a single problem, increasing the likelihood that these attributes would be in both result sets. To establish some kind of raw consistency between the results, it is important to determine just how many user test attributes corresponded to those found in the GSE models. In total, attributes were associated with the user test problem 22 times (or 22 boxes in Figure 5-48). However since no representation for User satisfaction (US) is provided or analysed in
any of the GSE models, making a direct comparison impossible, the 1 box, assigned
to US should be removed from the equation. Therefore, only 21 attribute instances are
taken into account. Since RS appears 4 times, and FG appear twice respectively, 8 of
the 21 boxes correspond to the attributes in the GSE problem, giving us a raw
consistency ratio of 38.09%. This is not surprising given the sum of the examiners'
comments tended to yield broader attribute results than the more formalised GSE
results.

<table>
<thead>
<tr>
<th>#</th>
<th>Problem</th>
<th>GSE Probs</th>
<th>Sub</th>
<th>Ex</th>
<th>Attributes</th>
<th>Pr%</th>
<th>ln%</th>
<th>Freq.</th>
<th>Severe</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Incorrect Code</td>
<td>G3</td>
<td>3</td>
<td>3</td>
<td>RS FG PR</td>
<td>100</td>
<td>38.1</td>
<td>RS</td>
<td>RS FG</td>
</tr>
<tr>
<td>2</td>
<td>Exiting Timer Record</td>
<td>V2</td>
<td>1</td>
<td>1</td>
<td>CT PR</td>
<td>0</td>
<td>0</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>3</td>
<td>Cancel to Correct</td>
<td>G6</td>
<td>3</td>
<td>3</td>
<td>ER EF DR</td>
<td>100</td>
<td>50</td>
<td>PR</td>
<td>DR ER</td>
</tr>
</tbody>
</table>

Table 5-6: Problems and Attribute Comparison for User Test Task 2 - G-Code

Thus, the result table in Table 5-6, for the GSE tasks, has a number in column 1
indicating the identifier to rank these problems found in the user test. Column 2
contains the name of the problem. Column 3 contains the GSE problems believed to
equate to this problem. Column 4 (Sub) marks how many of the subjects encountered
this problem in the protocols. Column 5 (Ex) records how many examiners considered
this to be a problem. Column 6 (Attributes) records the attributes identified in the
GSE problems listed in column 2. Column 7 Presence (Pr%) shows how many of
these attributes existed in the user problem. Column 8 inclusion (In%) compares how
many times these attributes were included in the user problem, against the total citing
of all attributes excluding user satisfaction. The attributes marked in bold in Column
6, are those that were found in both the GSE and user test results.

However, the attributes in Figure 5-48 are not consistently applied, and differ
significantly in frequency and severity. Column 9 (Freq.) records which attributes in
the GSE problem were reported as the most frequently occurring in the corresponding
user test problem. In the first user test problem ("Incorrect Code"), the attribute
responsiveness (RS) was one of the two most frequently occurring. This is equivalent
to the number of boxes in Figure 5-48 for RS. In the user test results CT was noted
with equal frequency, but since it was not found in the GSE result, it is not recorded
in this column.
Since 2 of the 3 attributes cited in the GSE problem (RS and FG) correspond to the highest severity in this problem (in this case category 1) they are contained in the "Severe" Column. This column records the attributes contained in both the user test and GSE results, which record the highest severity noted in the corresponding user test problem. If the highest severity recorded in this task had been 2, and RS and FG had at least one level 2 instance each, the result would be identical. Comparing the attributes contained in the frequency and severity columns to the total attributes cited in the GSE problems, it is apparent that 1 attribute (RS) out of 3 was listed as the most severe, and 2 out of 3, were listed as the most frequent. This gives us a comparative rating of 33% and 66.6% for frequency and severity respectively.

5.8.1 Overall User Test Results

Two out of the three examiners differed significantly in the number of problems. Examiner 1 marked 80 potential usability problems, Examiner 2 marked 104 problems and Examiner 3 marked a total of 88 potential problems. However given that many of the problems overlapped with one another in the transcripts, the total number of potential problems in the VCR study is 121.

Severity

All three examiners examined the 6 user test (think aloud) transcripts in detail in order to find the 121 potential usability problems and mark them for severity and relevant usability attributes. Severity is ranked from 1 (the most serious) to 4 (the least serious). The evaluators used the following table as a guide.

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Prevent Completion of a Task</td>
</tr>
<tr>
<td>2</td>
<td>Create significant delay and frustration</td>
</tr>
<tr>
<td>3</td>
<td>Problems have a minor effect on usability</td>
</tr>
<tr>
<td>4</td>
<td>Problems are more subtle and often point to an enhancement that can be added in the future.</td>
</tr>
</tbody>
</table>

*Table 5-7: Severity of usability problems, Dumas and Redish (1999:323-324).*

Not surprisingly, most problems were considered to be level 2 and level 3 severities. Category 1, a problem that impedes the completion of a task, was found only 7 times. Examiner 2 was unable to define any problems in this category. Category 2, any problem that causes significant delay and frustration, was the second most potential problem found. Category 3, problems considered to have a minor effect on usability,
was the most common. Examiner 3 considered this category to be extremely prevalent, by marking 73 out of 104 potential problems at this level of severity. The fourth and most subtle level of severity was the second least type marked. The allocation of severity to potential problems were:

<table>
<thead>
<tr>
<th>Examiner</th>
<th>0</th>
<th>Average</th>
<th>Studied</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>Sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>40</td>
<td>7.7</td>
<td>0.7261</td>
<td>2</td>
<td>31</td>
<td>16</td>
<td>11</td>
<td>80</td>
</tr>
<tr>
<td>2</td>
<td>48</td>
<td>2.7454</td>
<td>0.6626</td>
<td>0</td>
<td>29</td>
<td>73</td>
<td>2</td>
<td>104</td>
</tr>
<tr>
<td>3</td>
<td>48</td>
<td>2.6395</td>
<td>0.7201</td>
<td>5</td>
<td>30</td>
<td>42</td>
<td>9</td>
<td>116</td>
</tr>
</tbody>
</table>

Table 5.8: VCR Study Usability Problem Distribution

Attributes

The usability analysts also specified attributes, based on whether they thought a particular attribute corresponded to these potential usability problems. There was no limit to how many of these attributes were assigned to a given problem. To speed up this process the examiners used the codes in Table 5.9.

Two attributes, Configurability and Consistency are not identified by any of the examiners. While the lack of Configurability is not a surprising omission, given the scope of the study, the lack of Consistency attributes was surprising. A possible reason for this is that Consistency is a measure across all tasks, and unlikely to be identified on a case-by-case basis in a single transcript. An additional possibility is that Familiarity similar to Consistency, but meant to apply to knowledge of external

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Code</th>
<th>Summarised Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forgiveness</td>
<td>FG</td>
<td>Prevent Problems</td>
</tr>
<tr>
<td>Error Recovery</td>
<td>ER</td>
<td>Fix Problems</td>
</tr>
<tr>
<td>Responsiveness</td>
<td>RS</td>
<td>Quick Response Feedback</td>
</tr>
<tr>
<td>Controllability</td>
<td>CN</td>
<td>User in control – Can cancel actions</td>
</tr>
<tr>
<td>Directness</td>
<td>DR</td>
<td>Direct actions and visual objects assist user</td>
</tr>
<tr>
<td>Efficiency</td>
<td>EF</td>
<td>Tasks can be done rapidly and effectively</td>
</tr>
<tr>
<td>Learnability</td>
<td>LN</td>
<td>Easy to be learnt, does system support learning</td>
</tr>
<tr>
<td>Predictability</td>
<td>PR</td>
<td>Users can predict what the user will do in response</td>
</tr>
<tr>
<td>Simplicity</td>
<td>SM</td>
<td>Minimization of complexity, the less needed is more hidden</td>
</tr>
<tr>
<td>Transparency</td>
<td>TN</td>
<td>User does not need to know inner workings of system</td>
</tr>
<tr>
<td>Familiarity</td>
<td>FM</td>
<td>Interface is familiar to other external interfaces</td>
</tr>
<tr>
<td>Consistency</td>
<td>CS</td>
<td>Actions within interface are consistent applied</td>
</tr>
<tr>
<td>Configurability</td>
<td>CF</td>
<td>User can change settings, and also reverse them</td>
</tr>
<tr>
<td>Flexibility</td>
<td>FL</td>
<td>Different techniques for experts and novices are available</td>
</tr>
<tr>
<td>User Satisfaction</td>
<td>US</td>
<td>The user is satisfied with task (not frustrated)</td>
</tr>
<tr>
<td>Visual App / Layout</td>
<td>VA</td>
<td>Less used functions are hidden, more frequent ones are obvious.</td>
</tr>
</tbody>
</table>

Table 5-9: Attributes and Codes used in the User Tests
systems, is used instead. Another attribute Visual Appeal and Layout (VA) is not noted, except by one of the three examiners, and then only rarely. This was not unexpected given the examiners worked primarily with “think aloud protocols.” Table Figure 5-50 contains the frequency of the attributes assigned to potential problems by each examiner.

![VCR properties](image)

**Figure 5-50: VCR Pilot Study – Frequency of Attributes for Examiners 1, 2 and 3**

Directness (DR) and Predictability (PR) are the most common attributes identified by all three examiners. Responsiveness (RS) was the next most common according to both examiners 2 and 3. However, examiner 1 did not nominate Responsiveness. The next most common attribute is Learnability.

While the examiners differ in their attribute ratings, this is of no major concern, given the examiners did not identify the same number of problems, nor comment on them in exactly the same way. Given that usability problems in user tests are considered credible problems when one or more examiner agrees (see Chapter 4) because in user tests examiners do not always agree, it was anticipated that examiners would differ in the application of these attributes. Figure 5-50 shows how attributes were noted across the entire pilot study, and not for individual problems. So while giving us a broad indication of which attributes the examiners assigned to their respective sets of problems, it does not undermine the credibility of their findings when compared against those of the other two examiners. Instead, values concerning the severity and
frequency of attributes in section 5.8.2 to 5.8.7 better demonstrate consensus between the examiners.

**User Feedback**

At the end of the sessions, four of the six test subjects were asked to answer a number of questions about the VCRs used. It was interesting to find that each of these differed so significantly in relation to their favourite method of timer recording.

The feedback yielded some interesting results. The group was divided on which task was the easiest, with a different task for each subject. For Efficiency, task 1 the OSD procedure was considered to be the most efficient for 75% of the group. Subjects 2 and 3 believed that the Bar Code activity was the most difficult. Subject 1 considered the OSD to be the most difficult, in contrast to the other subjects who considered it to be the most efficient. When asked to indicate which VCR was the most difficult, the group was evenly split between the two VCR models. Though the sample size for the data above was small, it did demonstrate that in a comparative test, it is possible for the subjects to have very different views on their experience during the test.

**Top 3 Problems by average Severity.**

This analysis begins by identifying the most severe of the usability problems. These have been allocated a category 1 problem by at least 1 of the examiners. According to the list of potential usability problems, only 6 were identified as preventing the completion of a task.
The next section describes the problems encountered in each task. Charts containing the user test results corresponding to each task may now be found in the appendices.

5.8.2 Task 1 Results – Using the OSD

The OSD task, involves using the On-Screen Display that the VCR is able to present on a connecting television screen. This was accomplished via the remote control. The subjects had to perform six distinct activities by: entering three programs, checking them, correcting one of them, and finally cancelling them all. The actual instructions may be found in Figure 5-51.

<table>
<thead>
<tr>
<th>Task 1:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open Manual: Pages 24 – 25 Timer Recording</td>
</tr>
<tr>
<td>Program the following:</td>
</tr>
<tr>
<td>(Saturday): Channel Nine: 10:00 a.m. – 12:00 p.m. Football</td>
</tr>
<tr>
<td>(Monday – Friday): ABC: 7:00 p.m. – 7:30 p.m. News</td>
</tr>
<tr>
<td>(Every Sunday): ABC 12:00 p.m. – 1:00 p.m. Landline</td>
</tr>
<tr>
<td>Check all programs against above</td>
</tr>
<tr>
<td>Correct this program &lt;1&gt; Saturday Channel Nine 10:00 a.m. 12:30 p.m. Football</td>
</tr>
<tr>
<td>Cancel All programs</td>
</tr>
</tbody>
</table>

Description of the User Test Problems

In Problem 1, “Delete to Correct” there was some frustration, that in order to correct an incorrectly entered timer programme, or to change an existing one, all the previous data had to be lost. Learnability and Predictability were high, indicative of frequent manual references, and the VCR not behaving as expected. Problem 2 occurred when the user is unable to perform any operations on the VCR while it is locked into timer record mode. This primarily involved Predictability, Responsiveness given the VCR did not act as expected nor responded adequately. Problem 3 occurs when a user tries to set a timer programme, but is unaware the clock is not yet set. This means the VCR will record at a time and date different to that intended. Responsiveness is cited indicating a lack of feedback, and Directness possibly indicating a lack of opportunity to set the clock from the timer record screen.

Problem 4 corresponds to the difficulty involved in correcting a timer record value once it has already been entered. The high Directness result indicates that correcting the timer record values was not a straightforward task. It took a long time to learn and the manual was of limited value. Additionally the process is unfamiliar, unpredictable and provided poor responses as to the state of the system. The fact that level 1 was
assigned to Directness, Familiarity, Predictability and Responsiveness, indicates that the lack of these attributes prevented successful completion of the task. This problem is closely associated with “the delete to correct” problem since both issues impede the correction phase of task 1. In Problem 5 the user has difficulty specifying a weekly recording such as recording every Sunday. Users must cycle through all the days in a month, then all the preset ranges (e.g. Friday to Saturday) before getting to the days in the week. It is not surprising that this problem is largely associated with Directness, since entering a single value can take numerous button presses, predictability, as this is an odd and cumbersome way of entering values, and user satisfaction since this is likely to be frustrating, match this problem reasonably well.

**Comparison to GSE model**

<table>
<thead>
<tr>
<th>#</th>
<th>Problem</th>
<th>GSE Probs.</th>
<th>Sub</th>
<th>Ex</th>
<th>Attributes</th>
<th>Pr%</th>
<th>In%</th>
<th>Freq.</th>
<th>Severe</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Delete to Correct</td>
<td>O12, O13</td>
<td>2</td>
<td>3</td>
<td>DR EF ER</td>
<td>100</td>
<td>27.7</td>
<td>DR EF ER</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Stuck in VCR Mode</td>
<td>V2</td>
<td>1</td>
<td>3</td>
<td>CT PR</td>
<td>50</td>
<td>25</td>
<td>PR</td>
<td>PR</td>
</tr>
<tr>
<td>3</td>
<td>Clock not set</td>
<td>O1</td>
<td>1</td>
<td>2</td>
<td>RS ER FG PR</td>
<td>25</td>
<td>50</td>
<td>RS</td>
<td>RS</td>
</tr>
<tr>
<td>4</td>
<td>Trouble connecting</td>
<td>O12, O13</td>
<td>1</td>
<td>3</td>
<td>DR EF ER</td>
<td>33.3</td>
<td>40</td>
<td>DR</td>
<td>DR</td>
</tr>
<tr>
<td>5</td>
<td>Trouble Entering Sunday</td>
<td>O4</td>
<td>1</td>
<td>3</td>
<td>DR EF</td>
<td>100</td>
<td>44.4</td>
<td>DR</td>
<td>DR EF</td>
</tr>
</tbody>
</table>

*Table 5-12: Problems and Attribute Comparison for User Test Task 1 - OSD*

The GSE models were reasonably effective at identifying the same problems as those identified in the user tests. 4 out of 5 user test problems were identifiable in the GSE models for the OSD tasks. However only 5 GSE problems (O1, O4, O12 and O13) were needed to describe these. For most of these results they were identifiable in one subject’s talk aloud protocol only. However, in 4 of the 5 instances the examiners all agreed a potential problem existed. This satisfies the conditions specified in Chapter 4, that a problem is a clearly a problem when reported by 1 or more examiners.

The user tests again reported more attributes than those cited in the GSE results, the GSE attributes making up at best 50% of all attributes in each user test result. However in all 5 examples, at least one of the most severe attributes is consistent in both result sets. Problems 1 and 2, demonstrate that multiple attributes in both results sets were marked at the highest severity. 80% of the GSE problems reported one of most frequently occurring attributes. Directness was the most frequently cited attribute reported in both results sets. The task 1 results indicate that the GSE models
describe all the main problems in the user test results. The second finding is that though the user test results are much broader in their use of attributes, the severity and frequency comparison demonstrates significant consensus as to when attributes come into play.

5.8.3 Task 2 Results – Using G-Codes

The G-Code task involved using the G-CODE entry screen (part of the on screen display). In the G-Code Task, users had to do the same activities as task 1. However, the users had to consult an accompanying television guide to obtain the G-Codes corresponding to the programs specified in the tasks. As this task was conducted over consecutive days, some of the users had codes that were now “out of date” for programs that had already been aired.

Description of the Problems

In Problem 1 the user enters an out of date G-Code, and the system returns an error message stating the number is incomplete causing the user to think that they had not entered it correctly and try again. Controllability and Responsiveness are reported frequently indicating the inadequate response, and the user being unable to complete the task. Problem 2 involves the user exiting timer record mode before being able to cancel a timer recording. This was reported as a low severity problem given that once the timer-recording mode is understood the problem is unlikely to be repeated. Directness and Learnability are cited, since this impedes the completion of the task, and requires some manual references to learn what is going on. The last problem is similar to problem 1 of the OSD tasks, as the process for cancelling a timer programme is the same. The attributes reported are reasonably similar, except that flexibility is reported given that there is no alternate way to perform this function.

Comparison with GSE problems

The comparison table can be found in Table 5-6. 2 out of the 3 problems were detectable in the GSE models of the task. However task 2 is directly comparable with
the same problem in the VCR panel tasks reported later. Despite this, there was no consensus on the attributes between the results for problem 2 and those in the GSE models. 100% of the attributes in problems 1 and 3 were reported in both result sets, though only making up half of what was reported in the user test results. A positive finding was that the 2 out of the 3, attributes in the GSE instances were amongst those rated the most sever in the user test results, and responsiveness and predictability were found to be the most commonly occulting attributes in the user test results for problems 1 and 2 respectively.

The results for task 2, again demonstrate significant consensus on the core attributes involved in the problem, although the user test results are less specific than the GSE results. The lack of consensus between the attributes in problem 2 is disappointing but the fact that once again a majority of the user test problems were found in the relevant GSE models is a positive result none the less.

5.8.4 Task 3 Results Using the VCR Panel

<table>
<thead>
<tr>
<th>Task 3: VCR Panel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open Manual Pages 24 – 25 Timer Recording</td>
</tr>
<tr>
<td>Set timer recording for the following:</td>
</tr>
<tr>
<td>1. (Saturday): Channel Nine: 10 00 a.m. – 12 00 p.m</td>
</tr>
<tr>
<td>2. (Monday – Friday): ABC: 7.00 p.m. – 7.30 p.m News</td>
</tr>
<tr>
<td>3. (Every Sunday): ABC 12.00 p.m. – 1.00 p.m. Landline</td>
</tr>
<tr>
<td>Check Programs</td>
</tr>
<tr>
<td>Correct: Correct: &lt;1&gt; Saturday Channel Nine 10 00 a.m. 12.30 p.m. Football</td>
</tr>
<tr>
<td>Cancel All Programs</td>
</tr>
</tbody>
</table>

Figure 5-53: VCR Panel instructions

Entering the data using the VCRs display panel (fixed to the front of the VCR) is undoubtedly the least sophisticated of all the methods. This is the only task not to use a remote control in any way. Thus subjects had to sit close to the device to use this method. All the output was presented via lights and digital clock-like numbers on the display. On this device (as on the other) the timer recording information is shared by each task meaning that there is no separate repository for timer recording programmes entered by the remote control, VCR panel or bar codes.

Description of the User Test Problems

Though no critical (level 1) problem was noted for this task, meaning no problem was insurmountable; many less serious problems were yielded. Problem 1 happens when a subject cannot undo an unintended deletion, and Forgiveness and Directness were
both specified. In the second problem, to correct a timer programme, users had to delete and then re-enter the data. This was primarily associated with Controllability and Predictability, but several other attributes are mentioned, although surprisingly Efficiency is not noted. In the third problem, timer record mode (which locks out many of the controls) affected Controllability, Predictability, Learnability and Responsiveness, and User satisfaction, and unlike other instances Simplicity also. The fourth problem that was low in seriousness, again related to timer record mode, how it activated automatically, what it was and how to get out of it. This affected Directness, since it added an extra step to the task, Predictability due to its unexpectedness and had a mild effect on User Satisfaction. The fifth and final problem for the task, involved a simultaneous two-button press (+ and -) in order to cancel a timer programme. Several subjects expressed dissatisfaction. Efficiency, Simplicity, Learnability (lack of support) and Directness were all factors. Flexibility was also noted suggesting a lack of alternative methods to cancel a program, although it is possible to do the same thing from the remote or bar code scanner.

<table>
<thead>
<tr>
<th>#</th>
<th>Problem</th>
<th>GSE Probs.</th>
<th>Sub</th>
<th>Ex</th>
<th>Attributes</th>
<th>Pr%</th>
<th>In%</th>
<th>Freq.</th>
<th>Severe</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Can’t Undo</td>
<td>V4</td>
<td>1</td>
<td>1</td>
<td>DR RS FG ER</td>
<td>100</td>
<td>100</td>
<td>FG DR</td>
<td>FG DR</td>
</tr>
<tr>
<td>2</td>
<td>Cancel to Correct</td>
<td>V7</td>
<td>2</td>
<td>3</td>
<td>FG</td>
<td>100</td>
<td>13.3</td>
<td>FG DR</td>
<td>FG DR</td>
</tr>
<tr>
<td>3</td>
<td>Escape Timer Recording</td>
<td>V2, V3</td>
<td>3</td>
<td>3</td>
<td>CT PR, CS</td>
<td>66.7</td>
<td>45.4</td>
<td>PR</td>
<td>CT PR</td>
</tr>
<tr>
<td>4</td>
<td>Don’t Understand Timer</td>
<td>V2, V3</td>
<td>1</td>
<td>3</td>
<td>CT PR, CS</td>
<td>33.3</td>
<td>33.3</td>
<td>PR</td>
<td>PR</td>
</tr>
<tr>
<td>5</td>
<td>Two Button Press</td>
<td>V4</td>
<td>2</td>
<td>3</td>
<td>DR RS FG ER</td>
<td>25</td>
<td>25</td>
<td>DR</td>
<td>DR</td>
</tr>
</tbody>
</table>

Table 5-13: Problems and Attribute Comparison for Task 3 – VCR Panel

Comparison to GSE findings

All 5 User Test problems were identified in the GSE results, with only 5 GSE problems needed to describe the issues manifested in the tests. Problem 1 was one of the few instances when the GSE problems noted more attributes than the user test results. 100% or 2 out of 2 attributes identified in the user tests were predicted consistently. Clearly responsiveness and Directness are factors, however a lack of an undo function and feedback does imply that Error Recovery and Responsiveness (in the GSE results) are involved. Attribute rates for problems 2 to 5 demonstrate a low inclusion rate, showing once again how broadly attributes are applied in the user test results. However, the most frequently occurring attributes, as well as the highest severity attributes are reasonably consistent for problems 1,3,4 and 5.
The only attribute cited as the cause of problem 2, Forgiveness is recorded in both record sets. This is a reasonably concise description of the problem, although the user test attributes demonstrate other attributes that could conceivably be violated (including Familiarity). 2 out of the 3 attributes cited in the GSE problems associated with problem 3, Controllability and Predictability were both identified in the user tests as amongst the most severe. Though Forgiveness was cited in user tests it does not seem to be a factor in this problem, although Learnability and the lack of learning support is justifiably noted. Only Predictability was a common factor in the user test and GSE results for problem 4. Directness was the only attribute cited in both result sets for problem 5. While Flexibility (raised in the user tests) is appropriate given the lack of an alternate method Simplicity and Learnability do not stand out as root causes for this problem.

5.8.5 Task 4 Results Using the Remote Control

<table>
<thead>
<tr>
<th>Task 4: Remote Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open Manual Pages 26 Using the Remote Controller</td>
</tr>
<tr>
<td>Set timer recording for the following:</td>
</tr>
<tr>
<td>1. (Thursday): Channel Nine: 11 30p.m – 12.45a.m Twilight Zone</td>
</tr>
<tr>
<td>2. (Thursday-Friday): Channel Seven: 11 00 a.m. – 11 30 a.m. Ricky Lake</td>
</tr>
<tr>
<td>3. (Every Saturday) SBS 1.00 pm – 2.00 p.m. Journal</td>
</tr>
<tr>
<td>Check Programs</td>
</tr>
<tr>
<td>Correct &lt;!&gt; (Thursday): Channel Nine: 11.30p.m – 12.30. a.m. Twilight Zone</td>
</tr>
<tr>
<td>Cancel</td>
</tr>
</tbody>
</table>

Figure 5-54: Remote Control instructions

While both VCRs in the study had a remote control, the remote used on the SD20 is significantly different. First and foremost, the user entered details into the remote, which were stored and displayed on a small LCD at the top of the device. Only when the user had finished entering data, did they commit the programme to the VCR by pressing the transmit button. On the VCR used in tasks 1 and 2, any button press on the remote directly affected the VCR.

Description of User Test Problems

Problem 1 involved subjects specifying a two-day recording from Thursday to Friday, but because no preset exists, they have to enter Thursday and Friday separately. While no critical (level 1) rating was recorded, this problem was considered to be inefficient, inflexible and to cause significant user dissatisfaction. In problem 2, a built in bar code scanner (switched on and off via a switch on the remote) prevents users transmitting data to the VCR when it is switched on. Predictability (the transmit
button not working) and Responsiveness (no feedback) were key factors. In problem 3 users had trouble directing the VCR to record "every Saturday" because they either did not understand that 'S' signified Saturday in the date display or the clock not been set prevented progress. Predictability, Directness and Responsiveness were key factors, but Forgiveness (no learning support) and Forgiveness were also factors. In problem 4, subject Z had difficulty with the timer record process. Examiners noted unfamiliarity with previous models, Predictability, Forgiveness and Directness as factors. Problem 5 is the same problem noted in earlier tasks where timer record mode locks out all other functionality. This was a serious but not impassable problem involving Controllability, Learnability and Responsiveness.

**Comparison with GSE findings.**

In this task, only 1 of the 5-user test problems was directly associated with an instance, in the GSE models for the remote control. However, problems 3 to 5 were all consistent with problems described in the VCR panel tasks. Problem 1 which was not recorded at all, involves a significant amount of problem solving on the user’s behalf since they need to realise they cannot enter a multi day recording like they have in previous tasks. While this problem could probably be identified if the GSE models described the goals and methods for this task precisely this problem was directly identified in the study. Problem 2 however was identified, although only 2 out of 6 of the attributes were found in both result sets. Problem 3,4 and 5 were found in the VCR panel tasks (so not specifically the remote models). The frequency and severity comparisons are reasonably good, 1 out of 1 was high in severity for problem 3 and 1 out of 2 for problems 4 and 5. The presence of GSE attributes in the user-test results was also moderately good (50% and up) except for problem 2. On the whole these results again indicate that the user test results are broader in the assigning of

<table>
<thead>
<tr>
<th>#</th>
<th>Problem</th>
<th>GSE Probs.</th>
<th>Sub</th>
<th>Ex</th>
<th>Attributes</th>
<th>Pr%</th>
<th>In%</th>
<th>Freq</th>
<th>Severe</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Can't Do Thursday Friday</td>
<td>Task</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Scanner On / Off</td>
<td>R1, R2</td>
<td>1</td>
<td>3</td>
<td>DR FG PR ER, SM FM</td>
<td>33</td>
<td>43.7</td>
<td>PR</td>
<td>PR</td>
</tr>
<tr>
<td>3</td>
<td>Can't key in Saturday</td>
<td>V6</td>
<td>1</td>
<td>3</td>
<td>DR</td>
<td>100</td>
<td>14.2</td>
<td>DR</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Can't understand process</td>
<td>V2</td>
<td>1</td>
<td>2</td>
<td>CT PR</td>
<td>50</td>
<td>33.3</td>
<td>PR</td>
<td>PR</td>
</tr>
<tr>
<td>5</td>
<td>How to Cancel Timer</td>
<td>V2</td>
<td>1</td>
<td>3</td>
<td>CT PR</td>
<td>50</td>
<td>33.3</td>
<td>CT</td>
<td>CT</td>
</tr>
</tbody>
</table>

**Table 5-14: Problems and Attribute Comparison for User Test Task 4 - Remote**
attributes, but that the GSE models demonstrate reasonable consistency in frequency and severity.

### 5.8.6 Task Results Using the Bar Code Scanner

<table>
<thead>
<tr>
<th>Task 5:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open Manual: Pages 34 – 35 Channel Nine: 11.30p.m – 12.30a.m Twilight Zone</td>
</tr>
</tbody>
</table>

Set timer recording for the following:
1. (Thursday): SBS 9.00p.m – 9.15pm World News
2. (Thursday - Friday): Channel Seven: 4.00p.m – 4.30 p.m. Big Arvo
3. (Every Thursday) Nine 9.30 a.m – 11.00. a.m. Mornings with Kerri-Anne

Check Programs
Correct: <-1> (Thursday): SBS 9 00p.m – 9.30pm World News
Cancel All Programs

**Figure 5-55: Barcode Scanner instructions**

The last of the five tasks involved the barcode reader on the VCR’s remote, the same remote used in task 4. An accompanying bar code sheet was provided which the users had to scan using a scanner found at the bottom of the remote. There are significant problems experienced during the task, not the least of which is the scanning process itself, which proved difficult.

**Description of Problems**

The bar code task yielded the most problems identified in the user tests. In problem 1, the scanner on-off setting locked out the VCR controls, like they did in the last task. This was considered to be a severe problem with recurrent level 1 ratings. Problem 2 is equivocal to problem 1 in the remote control task, and since it was largely dependant on the task instructions was marked as an instructional problem. Problem 2, which involved having to start all over again to retransmit information to the VCR, was clearly inefficient, although the examiners picked Controllability and User Satisfaction as the primary attributes involved. Problem 4 was similar to 3, given that the user must start all over again to correct an entry if they make a mistake. The examiners proposed Controllability, Directness and Efficiency, but they did not note Forgiveness, which was surprising given the relationship of this problem to human error.

The fifth problem, involves the rapid speed at which the scanner must be moved along a bar code for it to register. No information in the learning material or as feedback lets the user know this is the case. Controllability, Forgiveness and Learnability were all marked as critical issues. The sixth problem which was considered less serious, occurs...
when codes are scanned right to left and don’t register. Again this is not mentioned in
the manuals or as feedback. Learnability and Flexibility are accurately cited, although
Directness and predictability seem less important. In the seventh problem, there is no
way to check a value once it has been scanned in. Responsiveness (lack of feedback)
and predictability (an expectation the value would be displayed somewhere) are
appropriately chosen here as are Learnability (the difficulty learning it) and
Directness. The eighth and final problem occurs when the subjects can’t find the
scanning light on the remote, and therefore can’t scan. This is a physical design issue
that could not conceivably be modelled in the GSE notation as modelled in this study.

Comparison with GSE findings,

<table>
<thead>
<tr>
<th>#</th>
<th>Problem</th>
<th>GSE Probs.</th>
<th>Sub</th>
<th>Ex</th>
<th>Attributes</th>
<th>Pr%</th>
<th>In%</th>
<th>Freq.</th>
<th>Severe</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Can’t Move Up Down</td>
<td>B3, B6</td>
<td>1</td>
<td>3</td>
<td>CT PR</td>
<td>50</td>
<td>33.3</td>
<td>PR</td>
<td>PR</td>
</tr>
<tr>
<td>2</td>
<td>Can’t Enter Thursday to Friday</td>
<td>Task</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Start Again to Retransmit</td>
<td>B5</td>
<td>1</td>
<td>2</td>
<td>DR EF CS FM</td>
<td>50</td>
<td>100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Cancel to Correct</td>
<td>B5</td>
<td>2</td>
<td>3</td>
<td>DR EF CS FM</td>
<td>50</td>
<td>33.3</td>
<td>DR</td>
<td>DR</td>
</tr>
<tr>
<td>5</td>
<td>Too Slow to scan</td>
<td>B4</td>
<td>1</td>
<td>3</td>
<td>RS LN ER</td>
<td>33</td>
<td>28.6</td>
<td>LN</td>
<td>LN</td>
</tr>
<tr>
<td>6</td>
<td>Can’t Scan Backwards</td>
<td>B4</td>
<td>1</td>
<td>3</td>
<td>RS LN ER</td>
<td>33</td>
<td>20</td>
<td>LN</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Can’t Check the Codes</td>
<td>B5</td>
<td>2</td>
<td>3</td>
<td>DR EF CS FM</td>
<td>25</td>
<td>22.2</td>
<td>DR</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Can’t Find Scanning Light</td>
<td>Physical</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5-15: Problems and Attribute Comparison for User Test Task 5 - Barcode

The GSE models were successful at predicting many of the problems in this task.
Excluding problem 2, which had a lot to do with the user problem solving a solution,
and problem 8, which was dependant on the physical design of the remote, 6 out of
the 6 or 100% of the remaining problems were detected. Again, only 4 GSE problems
(B3, B4, B5 and B6) were needed to find them. That said, the presence rates 50% in
50% of cases, and 33% or less in the remaining cases indicate differences between the
GSE analysis and user test results on which attributes apply. Finding the most severe
attributes was reasonably successful with 5 out of the 6 detected problems detecting 1.
While finding the most frequent attributes was consistent in 50% of the GSE
problems.

5.8.7 Overall GSE User Test Comparison

By analysing the user test results, 26 distinct problems were identified. Of these, 18
user test problems, or 69.02%, corresponded to GSE models of the task they were
identified in. This refers to task 1 problems that matched task 1 GSE problems, task 2 problems that matched task 2 GSE problems and so on. 3 problems 11.53% were not identified in the GSE models. One of these was considered to be dependant on the physical design of the remote control (the location of the scanning light). The other two corresponded to deceptive instructions, which instructed the subjects to set a recording range of Thursday to Friday, which was not a preset recording range, but had to be entered separately for each day. 5 of the problems (19.23%) however, are predicted in the GSE models for another task. In each case, these corresponded to GSE models of the VCR Panel task (task 3), which modelled the set-clock and timer record modes sufficiently well to describe these problems. This meant that 88.46% of problems were modelled in the GSE models (in at least one task model). This is a good result for the GSE models; given this is the first time they have been compared with user test results.

The average number of subjects for all problems was a low 1.58 with 15 problems found in the protocols applying to one subject only. However the number of examiners for a given problem was much higher (2.69), and only 2 of the 26 problems were noted by 1 examiner, 4 by 2 examiners, and the remaining 21 by all three examiners. This fulfils the prerequisite that a usability problem is clearly a problem if it is found by more than one examiner or more than one subject. Only two problems (one examiner and one subject) did not meet this requirement. However, they have been included such that a more extensive comparison with the GSE models could be made.

The number of attributes cited in the user test results was in the vast majority of cases, larger than those identified in the GSE problems. Across all problems (minus those not identified and not including User Satisfaction which is not predicted) an inclusion rate of 37.13% is calculated, meaning the user test results were far more generous with the allocation of attributes than the GSE models. This is not surprising given the user test problems were a composite of many instances across several examiners. The average presence of the GSE attributes in the user test results worked out to be approximately 56.85%, meaning at least half the attributes cited as the cause for a given problem in the user tests were also noted in the GSE results. Of the 23 user test problems identified, 86.96% identified an attribute considered to be the most severe,
and 69.56% of the GSE models predicted an attribute considered to be the most frequent in a user test problem. This demonstrates some reasonable consensus between the two result sets on which attributes were the root cause of a problem. An important point to note is which attributes used in the GSE models to predict problems were also identified in the user tests. Directness (10 out of 10) and Learnability (2 out of 2) are found in the user test results for every GSE model they are reported in (i.e. 100%). Efficiency (5 out of 7) and Responsiveness (3 out of 4) are found in the user test results most times they are reported in the GSE model. Error Recovery (2 out of 7), Controllability (2 out of 7) and Forgiveness (2 out of 6) are found in a minority of the user test problems corresponding to the relevant GSE models. And Consistency though cited in the GSE models 5 times but is never cited in the user test results. This demonstrates the attributes classified by the GSE problems are not always consistent with those proposed by the examiners.

An interesting observation is that only 18 GSE problems are required to identify the 23 problems in the user tests. This suggests that perhaps the patterns of behaviour (for finding problems) in the VCR GSE models might be better defined to apply to models involving other devices. The attribute results suggest further refinement is needed to make those attributes less consistently applied or less frequently used more accurate. However the pilot study was reasonably limited in its list of functions. The main study in the next chapter covers far more interaction methods than the number contained in this pilot study and should be more indicative of how successfully the attribute definitions are applied. However the high ratio of problems found by the GSE models, is a positive outcome for the method.

5.9 Summary

That concludes the user test results for all five tasks across the two VCR models. The results have demonstrated that most of the key problems can be found by analysing the models with GSE attributes. Table 5-16 contains a list of the problems detected in the user tests and references to their corresponding problems in the GSE models.

This chapter has introduced a transformation between NGOMSL and GSE, in order to design and evaluate a system on the basis of their goals. This pilot study has also
demonstrated that several attributes that contribute to usability can be determined, before deployment. The AS-IS representation was reasonably complex and time consuming, but this was not unexpected given the number of representations.

<table>
<thead>
<tr>
<th>VCR</th>
<th>Task</th>
<th>Problems in user tests</th>
<th>GSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>NV-HD60</td>
<td>OSD</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>O1</td>
<td>Delete to Correct</td>
<td>O12, O13</td>
</tr>
<tr>
<td></td>
<td>O2</td>
<td>Stuck in VCR Mode</td>
<td></td>
</tr>
<tr>
<td></td>
<td>O3</td>
<td>Clock Not Set</td>
<td>O1</td>
</tr>
<tr>
<td></td>
<td>O4</td>
<td>Trouble Correcting</td>
<td>O12, O13</td>
</tr>
<tr>
<td></td>
<td>O5</td>
<td>Trouble Entering Sunday</td>
<td>G4</td>
</tr>
<tr>
<td></td>
<td>G-Code</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>G1</td>
<td>Incorrect Code</td>
<td>G3</td>
</tr>
<tr>
<td></td>
<td>G2</td>
<td>Exiting Timer Record</td>
<td></td>
</tr>
<tr>
<td></td>
<td>G3</td>
<td>Cancel to Correct</td>
<td></td>
</tr>
<tr>
<td>NV-SD20</td>
<td>VCR Panel</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>V1</td>
<td>Can’t Undo</td>
<td>V4</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td>Cancel to Correct</td>
<td>V7</td>
</tr>
<tr>
<td></td>
<td>V3</td>
<td>Escape Timer Recording</td>
<td>V2, V3</td>
</tr>
<tr>
<td></td>
<td>V4</td>
<td>Don’t Understand Timer Record mode</td>
<td>V2, V3</td>
</tr>
<tr>
<td></td>
<td>V5</td>
<td>Two Button Press</td>
<td>V4</td>
</tr>
<tr>
<td></td>
<td>Remote Control</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R1</td>
<td>Can’t Do Thursday / Friday</td>
<td>Task</td>
</tr>
<tr>
<td></td>
<td>R2</td>
<td>Scanner ON/OFF</td>
<td>R1, R2</td>
</tr>
<tr>
<td></td>
<td>R3</td>
<td>Can’t Key in Saturday</td>
<td>V6</td>
</tr>
<tr>
<td></td>
<td>R4</td>
<td>Can’t Understand Process</td>
<td>V2</td>
</tr>
<tr>
<td></td>
<td>R5</td>
<td>How to Cancel Timer Record</td>
<td>V2</td>
</tr>
<tr>
<td></td>
<td>Bar Codes</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>B1</td>
<td>Can’t Move Up Down</td>
<td>B3, B6</td>
</tr>
<tr>
<td></td>
<td>B2</td>
<td>Can’t Enter Thursday</td>
<td>Task</td>
</tr>
<tr>
<td></td>
<td>B3</td>
<td>Has to Start Again to Retransmit</td>
<td>B5</td>
</tr>
<tr>
<td></td>
<td>B4</td>
<td>Cancel to Correct</td>
<td>N5</td>
</tr>
<tr>
<td></td>
<td>B5</td>
<td>Too Slow to Scan</td>
<td>B4</td>
</tr>
<tr>
<td></td>
<td>B6</td>
<td>Can’t Scan Backwards</td>
<td>B4</td>
</tr>
<tr>
<td></td>
<td>B7</td>
<td>Can’s Check the Codes</td>
<td>B5</td>
</tr>
<tr>
<td></td>
<td>B8</td>
<td>Can’t find scanning light</td>
<td>Physical</td>
</tr>
</tbody>
</table>

Table 5-16 – Problems found in Pilot Study User Tests and GSE model equivalents where predicted

One factor that could have made this easier (and faster) might be access to an existing design model, and not a reliance on manual translation and observation. Automated tools, as in other methods may help this process, but these are not available at present.

Despite all this, building the GSE model and analysing it, involved less time and labour needed than to conduct the user testing and analysis, even though the user tests had been conducted on the minimum six subjects only.
This is especially true considering all the factors involved in preparing a user test including:

- Preparing the test material, and equipment
- Surveying and obtaining subjects
- Conducting and administering the test sessions for all six subjects.
- Processing feedback forms and surveys
- Transcribing think aloud protocols
- Expert analysis of protocols and identification of potential problems
- Examining these results, and identifying the core problems.

It would be inappropriate to only apply the methodology to devices with a limited set of functions. In order to overcome this, the next chapter describes the main study, which is conducted on modern mobile phone technology. While the tasks in the VCR user tests all dealt with an aspect of timer recording, (entering, checking and cancelling) the phone study is far more extensive. The next study also demonstrates how a NGOMSL model could be constructed as the basis for a future GSE model. However the primary aim of the main study is to determine how effectively usability problems can be identified in a GSE task model and therefore prevented.
CHAPTER 6
MOBILE PHONE STUDY: TASK MODELS

"Several usability challenges have emerged as a result of the new technological developments that have taken place. As devices have gotten smaller and more portable, designers and developers need to think about how users will be able to use these many new devices and develop new metaphors for them."

Rozanski, and Haake (2003: 184)

6.1 Introduction

The ongoing evolution of the mobile phone is nothing short of extraordinary. These ubiquitous devices can currently provide: real time video communication, multimedia recording, personal organising, wireless connectivity, hands free calling, Internet access and much more besides. Yet, somehow this new technology must also retain the traditional capabilities of the phone, such as making and receiving a call, within the constraints of a user interface limited by its own compactness. And still the mobile phone continues to evolve, as manufacturers compete to dominate the market by making their phones more and more versatile. The evolution of these compact interactive devices has significant implications for usability design.

This study is primarily is to test the ability to predict problems on the phone, using the attributes and the GSE based approach. However there are numerous mobile phone studies worth mentioning. These studies include:

- Addressing the physical design of the phone buttons (Mackenzie 2002 and Hirotaka, 2003).
- KLM models of text entry on Korean mobile phones (Myung 2004).
- Dissatisfaction with mobile phone design. (Nielsen 2004).
- How Nokia, one of the leading mobile phone manufacturers, invests in usability (Korhonen 2000).
  James, and Reischel, (2001) and more recently in Pavlovych and Stuerzlunger (2004).
The aim of this chapter is to determine the effectiveness of the NGOMSL and GSE modelling technique proposed in this study, when applied to the ubiquitous design of the modern mobile phone. Unlike the previous chapter, where VCRs were tested in relation to timer recording, this chapter discusses a much larger range of functions, on devices far more advanced than those in the previous study. Instead of relying upon a pre-determined NGOMSL model of the users goals, as in the last chapter, no such model exists for this study. Instead, an entire NGOMSL model must be built from scratch, since none are either available or suitable. Then As-is models of the actual tasks and built integrating based on some of the goals and sequences in the NGOMSL representations.

This chapter is divided into seven sections. The next section 6.2 documents the construction of a NGOMSL model of user’s goals in relation to mobile phone devices. This model is then transformed into a GSE representation using Kieras’s procedures for building NGOMSL models. In Sections 6.3 and 6.4 the two phone devices used in the study are transformed into “As-Is” GSE models, using the same techniques used to model the VCR devices in Chapter 5. The final section 6.5 summarises the findings of these chapters, before outlining the issues discussed in the results and discussions chapter.

### 6.2 The Mobile Phone NGOMSL Model

A full NGOMSL model is built in stages, much like a GSE model. A NGOMSL Model begins with a detailed representation of the user’s goals in natural language. The section documents the construction of a NGOMSL model for mobile phone users using the NGOMSL process as documented in Kieras (1997). This chapter is broken into several sections, which correspond to each stage of the NGOMSL process used to build the NGOMSL model. These stages can be seen in Figure 6-1.

![Figure 6-1: The NGOMSL Process in Full. Based on steps in Kieras (1997)](image-url)
The NGOMSL model used in this chapter is far larger than the VCR model in the previous chapter. Due to its limited focus, the NGOMSL VCR model was converted into GSE relatively simply. However the model constructed in this section, unlike the pilot study, covers a far more extensive range of methods, goals and operators. That was in part due to the limited capabilities of the VCRs themselves. In this chapter, however a large range of ubiquitous and traditional tasks are examined, in keeping with the wide application of both GSE and NGOMSL methods. To model these tasks, the first stage of the NGOMSL process is to determine the user’s top-level goals.

**Step A: Choose the top-level user’s goals**

Kieras (1997) recommends that the NGOMSL analyst should aim for high-level goals to avoid predetermining the structure of the model. But what are the users’ goals? The highest possible goal for operating a mobile phone, is to “Use or operate a phone,” since this describes any phone activity. To help the process, Kieras has provided two starting templates for building a NGOMSL model in Table 6-1

<table>
<thead>
<tr>
<th>Top Level Goal</th>
<th>Top Level Selection Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method for goal: Edit the Document</td>
<td>Selection rule set for goal: perform the unit task</td>
</tr>
<tr>
<td>Step 1: Get next unit task information from marked-up manuscript</td>
<td>If the task is deletion then accomplish goal: perform deletion procedure</td>
</tr>
<tr>
<td>Step 2: Decide if no more unit tasks, then return with goal accomplished</td>
<td>If the task is copying then accomplish goal: perform copy procedure ... etc</td>
</tr>
<tr>
<td>Step 3: Accomplish goal: Move to the unit task location</td>
<td>Return with goal accomplished</td>
</tr>
<tr>
<td>Step 4: Accomplish goal: perform the unit task</td>
<td></td>
</tr>
<tr>
<td>Step 5: goto 1</td>
<td></td>
</tr>
</tbody>
</table>

Table 6-1: Kieras’s (1997:749) Starting Templates for NGOMSL

These templates are clearly for text editing, however the structure is a useful guide to starting the phone model. To use a phone, there are two common sense factors that must first be addressed:

- The phone must be activated, before any phone function can be used.
- The user can only answer a call if a call is incoming.

Using these pre-conditions, and the template specified in Table 6-1, it is possible to build the method corresponding to the highest-level goal: use the phone:
Method for goal: Use Phone
Step 1: Decide: if phone is off, achieve goal: activate phone
Step 2: Decide: If call is incoming Achieve goal: respond to call.
Step 3: Decide: if no more tasks then return with goal accomplished
Step 4: Accomplish goal: Perform Phone Task
Step 5: Go to 1

The natural language of NGOMSL makes it easy to specify the preceding rule. The user decides if the phone is off, and if so activates the phone. If a call is incoming, the user responds to it. Note that this response may be ending the call, (not accepting it) so the phrase “responding to call” was chosen instead of “answering the call” which will be drafted later as a sub goal.

The second part of the template in Table 6-1 is a selection rule. Selection rules should be used when a long set of mutually exclusive choices exists. Once any of these tasks have been chosen, the others are unavailable for the time being. A subset of tasks was chosen for the study, a collection of ubiquitous functions (e.g. games, calculator, camera) and more traditional functions such as making a call, sending an SMS, using contacts. The tasks chosen, in keeping with Kieras’s’ guidelines, were specified at as high a level as possible, as the starting point. The phones discussed later in the user study, offered many of these capabilities, which is why they were chosen. However it is not possible to test “all” the functions of these phones in the user tests, and since the aim was to identify problems ‘before testing’ a subset of tasks were used and tested.

Core Functions
- Turning a phone on / and off
- Making a Call,
  - manually,
  - using speed dial,
  - from contacts,
  - using voice recognition
  - from SMS
- Creating and sending an SMS message
- Changing Contacts (create, deleting and editing)

Additions (Accessories)
- Setting a reminder
- Using a calculator
- Playing a Game

More Recent Additions
- Taking a photo or video sequence
- Connecting to the Web (Services and email)
Therefore each of the above tasks is included in the NGOMSL model built in this section. There are a number of important pre-conditions that are specified for this study, based on the standard physical interface of most mobile phones. The phones evaluated later in the study share all of these common features:

- Navigation Keys (Up Down Left and Right) are available
- Numeric Key Pad keys are available
- An on off button is available
- Select buttons are available
- Start Call and End Call buttons are available
- A menu button is available
- Phone has a menu system of some kind (and all functions except phone off and on can be got to by menu)
- Web connectivity uses standard HTML when connected

The next NGOMSL method to be built is Perform Phone Task. Since the tasks described in it are “mutually exclusive” and in keeping with Kieras’s template, a selection rule will be used instead of a method. The following is a first ‘pass’ of the perform phone task based on the assumption that each task is independent of the others.

**Selection rule set for goal: Perform Phone Task**

- If the task is making a manual call then accomplish goal: Make Call manually
- If the task is making a call with voice dial then accomplish goal: Make Call using voice dial
- If the task is making a call then accomplish goal: Make Call using speed dial
- If the task is making a call then accomplish goal: Make Call using call history
- If the task is using a text message then accomplish goal: create text message
- If the task is using a text message then accomplish goal: open text message
- If the task is send a text message then accomplish goal: send text message
- If the task is using an accessory then accomplish goal: use calculator
- If the task is using an accessory then accomplish goal: add reminder
- If the task is using an accessory then accomplish goal: delete reminder
- If the task is to edit contacts then accomplish goal: edit contacts
- If the task is to edit contacts then accomplish goal: download game
- If the task is access web application: the accomplish goal: send email
- If the task is access web application: the accomplish goal: access web services
- If the task is to take photo: then accomplish goal: Take Photo
- If the task is to take video recording then accomplish goal: Take Video
- If the task is to send a media file message then accomplish goal Send Media File

Return with goal Accomplished
This is a very large selection rule and is supposed to represent the user’s high-level goals. Looking at the users’ goals, the analyst is able to specify any assumptions. To revise this selection rule, a number of assumptions are to be made here. These are:

1. That all the methods for making a call, should be contained in the larger goal “make call”.
2. The user recognises the connectivity of the text methods (Create, Open and Send) and is able to identify that they all fit into a larger goal “Use Text Message”.
3. That the user recognises the connectivity of methods concerning multimedia. Therefore (the take photo, take video, and send media file) are all related and fall into a larger user goal “Use Media”.
4. That reminder, games and the calculator functions can be considered accessories and structured under a “use accessory” goal.

Assumptions are typically stated in underneath the first line of a method. For the new perform task (below) only a reference to the assumptions listed above is made.

**Selection rule set for goal: Perform Phone Task**

See Assumptions 1 -- 4 as above

- If the task is making a call then accomplish goal: **Make Call**
- If the task is using a text message then accomplish goal: **use text messages**
- If the task is using an accessory then accomplish goal: **use calculator**
- If the task is to edit contacts then accomplish goal: **Change contacts**
- If the task is to edit contacts then accomplish goal: **Play Game**
- If the task is access web application: the accomplish goal: **access web services**
- If the task is use media: then accomplish goal: **use camera**

Return with goal Accomplished

With the highest-level NGOMSL notation conducted, it is now possible to start drafting methods, from the top down. Step B “Do the following recursive procedure” is broken by Kieras into three steps, which can be repeated across several passes to ensure the validity of the model.

**6.2.1 NGOMSL Step B – The Recursive Procedure**

This second step of the NGOMSL procedure is a recursive procedure for building methods, checking them against assumptions and moving to new levels of detail if necessary. This is a cycle that must be repeated several times, in order to refine a
crude NGOMSL model to one that is complete, natural and consistent. These iterations are called consistent with Kieras’s (1997) documentation on the NGOMSL method. Due to the significant size involved in developing the model across several iterations or passes, this step is documented in detail in Appendix F.

6.2.2 Step C: Document and Check the Analysis

In this stage, the NGOMSL model must be examined in relation to both assumptions and the naturalness of the design. This process covered the following:

- **Primitive External Operators and Analyst Defined Operators.**
  Several external operators were implied such as Line up the camera and Browse web. Since neither were a critical part of the user study, no additional analysis was carried out, nor times assigned to any analyst defined operators.

- **Assumptions and Judgement Calls**
  Many of the assumptions in this model relate to the capabilities of the phone, and the knowledge of the user that these features exist. For instance, that the user is aware that certain devices are available. Since the User Tasks were provided with instructions, many of the assumptions on what they “know” about a task, can be determined from the task instructions and manual references. The subjects were all assumed to have used mobile phone models and not the ones in the study (refer to Chapter 4.)

- **The Contents of the task description for each task**
  Each of the methods already documented, in the later passes were tested by using the common sense question, mentioned in Kieras (1997) “Is this how a user would describe it?” Though some may be very formal, like “Enter number” these were deemed to be adequate though it would be equally correct to use “Enter a Phone Number”.

6.2.3 Step D: Check Sensitivity to Judgement Calls and Assumptions

- **Learning –** The users were assumed to be novices. However in the multimedia, task, “Accessing the media library” was as good a description as any. No learning times were calculated; in this user study since the primary aim was to assess the quality attributes against NGOMSL based models of predicted phone activity.
• Execution Time – Due to the inexperience of the users, and the vast number of tasks, execution time was not calculated, though KLM operators could be applied to many of the atomic button presses if necessary.

• Workload, the amount of knowledge needed to complete a task (i.e. working memory) is not analysed in great detail, since the primary aim is to find usability problems with the GSE modelled attributes based on the NGOMSL method structures.

6.2.4 The finalised NGOMSL Mobile Phone model

After conducting the NGOMSL process, the final NGOMS model consists of the following methods and selection rules:

Using the Phone – High level goals [3]
- Use Phone
- Perform Phone Task
- Activate Phone

Making Calls [10]
- Make a Phone Call
- Start a Standard Phone Call
- Choose Number
- Input number from SMS
- Input Number from Contacts
- Input Number from Recorded Calls
- Make Voice Activated Call
- Make a Speed Dial Call
- Input Number Manually
- Search Entries for Number

General Operations [8]
- Go to Destination
- Navigate Menu to Destination
- Pick Command from Menu
- Press a Key
- Enter Character
- Pick Digit
- Scroll to Destination
- Move Pointer to Icon

Text Message Functions [8]
- Use Message
- Open Message
- Perform Message Task
- Send Message
- Read an Open Message
- Open Message
- Save Message
- Edit Message

Text Entry Functions [3]
- Perform Text Function
- Move to Task Location
- Edit Text

Calculator [4]
- Calculate an Equation
- Get Answer
- Enter Equation
- Enter Operator

Reminders [8]
- Manage Reminders
- Perform Reminder Task
- Add Reminder
- Delete Reminder
- Choose Reminder
- Edit Reminder

Playing Games [3]
- Start a Game
- Find a Game
- Activate Game

Web Related Methods [4]
- Use Web Services
- Download Application Game
- Access Web Based Email Site
- Connect to Web

Multimedia Related Methods [6]
- Take Photo
- Take Video
- View Media
- Delete Media File
- Send Media File

Contact Related Methods [8]
- Add Contact
- Find Contact
- Delete Contact
- Create New Contact
- Edit Contact
- Enter Details
- Enter Text Field
- Delete Contact
These models, above are used as a source for many of the goals and operations conducted in the mobile phone tasks. Additionally, GSE representations for many of the NGOMSL models in a literal form (based on the formatting in this first section) can be found in the Appendices. This next section constructs GSE models based on the NGOMSL models in order to identify potential usability problems.

6.3 Building the NEC GSE Model

The NEC was chosen due to its extensive set of features. The phone is able to access the web, record videos, and hosts a large number of other features. These extensive ranges of capabilities make it an excellent example of ubiquitous technology. The physical layout of the phone is demonstrated in Figure 6-2.

Unlike traditional phones, the NEC supports Java applications. Since a finite number of java applications can be open at a single time, the phone is more complex in its operations than the other used in the study.

These issues are discussed later in this section. Since this section discusses the conversion of the NEC's behaviour to a GSE model, the first thing to do is to identify the components that will make up this model.

The phone is turned on by holding the end key (marked as H for hang-up) in Figure 6.1. After a few seconds the phone will display the start screen (see figure 6-2) which displays the time and date, the icon indicator which shows the status of the phone, and the access to the primary functions (accessing user contacts (and phone numbers as well as SMS text messages).
As in the VCR conversions to GSE in Chapter 5, it is essential to determine the system component. In this instance the system component is the phone itself. In keeping with GSE notation, this system component is drawn with a double border as shown in Figure 6-3

![Figure 6-3: NEC system component](image)

With the system component now specified, the other components should also be identified. The screen and buttons in figure 6.1 are all components in their own right. These keys are as follows:

- **Left (L) and Right (R) Soft Keys**
- **Numbered Keys** 1, 2, 3, 4, 5, 6, 7, 8, 9 and also # and *
- **Make Call and End Call Keys**
- **The Menu key**
- **The Cancel Keys**

These keys are essential to the interface of the mobile phone and are used frequently used in the GSE model of the phone.

**Screen Components**

There are two global graphical components, which are critical to the functionality of the phone, since they appear on almost any screen. These are the **Icon Indicator** and the **Prompt Bar**. These two components sit on the top and bottom of the standard screen display and are shown in the start screen in Figure 6-4. The **date** and **time** appear on the start screen, but unlike the icon indicator and prompt bar, are not

![Figure 6-4 The NEC Start Screen](image)

Page 256
displayed on the majority other screens. The Date and Time are noted as individual components in keeping with GSE notation.

The Icon Indicator contains several sub components, which correspond to the status of the phone in relation to a number of different functional areas. These icons notify the user that the alarm is active, phone vibration is on, the battery power level, the current signal strength and other bits of information relevant to the function of the phone.

![Diagram of Icon Indicator subcomponents]

**Figure 6-5: Subcomponents of the Icon Indicator**

Critical to the function and understanding of the phone interface is the prompt bar that sits at the bottom of the screen (see Figure 6-4). The prompt bar defines the current function assigned to several buttons on the phone. It has three main sub components. These are:

- **The Left Soft Box and Right Soft Boxes**, which correspond to the functionality of the left and right soft keys (L and R in Figure 6-2).
- **The Navigation Indicator**, which lets the user know whether the up, down, left, and right buttons are active.

The phone displays dozens of different screens, some of which may actually be updated or revised, through web connection to the service provider.
In GSE screens are generally represented by a display or screen component in a different state. In this as-is model the start screen is a state of the screen component, which is named as Display.

This is depicted in Figure 6-6. In this screen box, the Icon Indicator is dealt as one component. The prompt bar in broken into its subcomponents, the left soft box, the nav bar, and the right soft box, to make the state of these components clear. These will be demonstrated as the user tasks are converted into their GSE representation.

The AS-IS model is built in the same way as the VCR as-is models were constructed in the previous chapter. The instructions in the manual match reference numbers provided to subjects in the user study. Based on these instructions a literal translation of the phones behaviour across these tasks can be constructed. There are seventeen tasks in the VCR study, and so seventeen literal models of this activity are to be built and used as the basis of this model.

<table>
<thead>
<tr>
<th>Edit Message Screen and Sub Menu</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Settings Area changes according to what's being entered. Names have a 30-character limit, which will be shown in remaining characters. The submenu is consistently applied to the different text entry screens with only some variation</td>
<td></td>
</tr>
</tbody>
</table>

*Figure 6-7: Edit Screen: Diagram is adapted from content and screen captures in the 3 user manual.*
6.3.1 Task 1: Turning the phone on

It is important to stress that the numbering system used for the tags in the GSE models built in this section, is arbitrary. Some models have been numbered sequentially, such that recurring and equivalent nodes (those connected by the $^\wedge$ or $\alpha =$ notation) can be numbered identically. However this is not the case in all the models in this chapter.

The user’s goal for task 1 is to turn the phone on. Section 2.1 of the manual contains a set of instructions for doing so.

![Turning your handset on and off](image)

These have been formatted for GSE conversion using the same technique for the as-is VCR models in the previous chapter. These instructions are as follows:

User Goal 1: Turn the **Phone** [On]

**Preconditions:** **USIM** [in handset] and **Battery** is [charged].
- [Press and hold] the **end key** (On/Off) until the **power** is [on]
- [Press and hold] is equivalent to approximately two seconds
- **User** Press and [Hold] **End Key** for Two Seconds, Verify [On]

Additional Prerequisites: Phone must be off. The User must only release the button when the phone is on, which must occur after two seconds.
The revised version adds a display node to more accurately represent the process. After the phone is in [starting up mode], the start up screen appears. The potential problem is made a little more complicated by the fact that no actual duration is specified in the manual. In reality though, it takes about five seconds for the button to activate [1.1]. There is no system response until after this [1.3], meaning that Responsiveness is the single most obvious problem here. Directness, and Efficiency may also be affected by the delay.

**Problem Analysis.**

Only one problem was found.

**Problem N1a** is deemed to violate **Efficiency** (for the five second delay)

**Learnability** (since no time data is mentioned in the manual) and most importantly of all **Responsiveness**, since no response occurs while the button is held until it reaches five seconds. **Directness**, since any Directness is interrupted by the time delay and no visual cue provided.

**NGOMSL comparison** [Assumption – User is aware of the power button]

Familiarity. – This method is consistent with the NGOMSL goals and the Samsung model later. Because a display component appears before a Verify operator it is unlikely that Responsiveness is an issue in the later node.
6.3.2 Task 2: Making a phone call

Making a phone call is a larger task to represent compared to turning the phone on.
The user must enter a sequence of digits and then activate the call. Figure 6-10
contains the four main screens in the actual NEC sequence.

![Image of phone screens]

**Figure 6-10: NEC Task 2: Main Steps and screens in making a manual phone call.**

**Formatted Description: Making a Phone call from the Start Screen:**

After integrating the Manual instructions, observations and NGOMSL sequences, the formatted of the
task sequence is as follows:

The user’s **goal** is to activate a call. The **user** may decide to enter a number manually.
The **user** enters the first digit, which opens up the [Make Call] **Screen**. The **user**
enters the digits into the **phone number**, until the **user** is ready to activate a call. The **user**
can delete the last digit entered by pressing the **CLEAR key**, or Delete all the
digits in the **phone number** by [holding the] **Clear key** for [five seconds] To activate
the call, the **user** must press the **send key**; the **phone** then takes the **phone number**
from the screen and starts connecting.
Problem Analysis for NEC Task 2 Part 1

In the above diagram potential problems have been noted and tagged based on the attributes modelled in chapter 4. Problem N2a, N2b, and N2c are all mild Responsiveness problems (since no on-screen input defines the purpose of these particular buttons). However this is countered by Learnability and Familiarity, since the function of these keys is specifically defined in the procedures, and Familiarity, as these particular buttons are much like their counterparts on other phones. Several of the blue goal boxes above correspond directly to steps in the NGOMSL method make a phone call. All these stages can be carried out in the process, and its no surprise that the manual description doesn’t mention that the subject needs to actually converse, although that is clearly the purpose of the call. The GSE model, unlike NGOMSL is
able to represent screen content, though it is still possible for screens to be implemented in such a way that this content is still unreadable. However the above model demonstrates how system presentation and less well-defined user behaviour such as conversing can be unified in a single model.

Figure 6-12: NEC Task 3 Making a Call Part 2
Formatted Description for Part 2 (in Figure 6-12)

In the second part of making a call, the user must actually converse and then end the call, provided the phone can actually connect. The sequence is as follows:

**Screen** Indicates the phone [is calling]. If the Phone [Can’t Connect] An Error Message is displayed. If it [can connect] goes to screen [ringing]. If the phone [rings out], the number is [emptied], and the Phone is [not connected]. If the phone is [answered] the [talking to] screen opens, the user next goal is to [converse], The user must [decide to end call], by [pressing] the hang-up button. This [empties] the phone number completes the goal and returns them to the [start] screen. If user wishes to end call, while the phone is [ringing] if the head set is [in use] the user must press end button and [close] handset, the call is then [ended]. If the head set is not used the user [press] the end button, The phone call is [ended].

**Problem Analysis**

**Problem N2d** is an Efficiency problem, since the number is wiped immediately. It cannot be corrected without re-entering.

**Problem N2e** is a Simplicity violation, since it deviates from the basic NGOMSL goals substantially and Consistency, because two different methods apply to turning the Phone off.

Problem 2 Demonstrates how complicated a simple task sequence can seem when integrated with system components. NGOMSL is able to distinguish the task amongst the response of the system, but most of the problems in this task apply directly to the arrangement of the system components.

6.3.3 Task 3: Add following names and number to phone book (contacts)

In this task the users had to enter the following bits of information into the Stored Contacts (Names and Numbers) of the NEC:

- Name: Al, Number: 0407 515 530 to phone book (contacts)
- Name: Al, Number: 3848 2885 to phone book (contacts)

There are a number of NEC quirks that constrain the functionality in this task. A description of this task is broken into three parts, Choosing a data source, naming a contact and creating the contact.
Part 1: Choosing a Data Source

First the task begins by holding down the right soft key, while in the start screen since the right soft box contains the Contacts option. This opens a "Destination Screen" where users may choose which of two data sources may be used to store the data. There are two sources in the NEC for contact information. The first is the USIM (the Card) and the actual phone memory itself. There are significant differences in relation to how much data can be stored.

Part 2: Naming a Contact

Once the destination is chosen the process continues on to a name input screen. The name is entered here and then the user must then press the ENTER key to continue.

Part 3: Creating the Contact

Finally the User actually arrives at the Add Contact Screen. How much data can be stored is dependant on the actual destination for the data and so there are essentially two add contact screens. The USIM Add Contact Screen has only four fields, while the Add Contact Screen for Phone based contacts in Figure 6-15 is actually split over two pages and offers a whole range of other items such as ring tone and picture settings.
Based on the NGOMSL models for setting contacts the relevant NGOMSL goals for this task are Add Contact, Access Contacts, Create New Contact, and Save Contact. Taking the instructions from the manual reference for this task and adding the NGOMSL goal structure the following text is the basis for the NGOMSL-GSE model.

**Formatted Description**

The Formatted Description of the task is as follows:

User’s goal is to add contact. From the start screen the User must [Press and hold] **Contact** for [5 seconds]. They must then scroll to and select **Phone Contacts List** or **USIM Contacts**. They must then select Create new Contact.

**Problem Analysis**

**Problem N3a**: is a Predictability, and Simplicity problem. Holding the button to get to the destination screen and having to hold it less to go through the Menu is not immediately apparent.

**Problem N3b**: is a Directness and Familiarity issue. Is Phonebook in user data?

**Problem N3c**: Is a Transparency issue and is an issue not even considered in the relevant NGOMSL models for adding a contact. On the NEC the user must pick a data source, the USIM or the Phone. While this demonstrates Configurability, it does add additional detail to the task.

**Problem N3d**: is a Forgiveness issue. The phone will accept a blank name, and even a duplicate as an index to the phone records. This need not be known to detect this problem. The fact the process accepts the data without any checks on it demonstrates this problem.

**Problem N3e**: is a Transparency and Predictability problem, since the behaviour branches off to obtain two different sources of data. The content difference between the records is not modelled here, but a potential Consistency problem emerges in relation to the structure of the data.
Figure 6-16: Add Contact Screen for Phone-Contacts Part 1

Formatted Description of NEC Task 3 – Part 2

Assuming the user has chose Phone Contacts.

The goal is to [Enter Name]. The user must Key in the contact name and [press] the enter key. This opens the [details] screen. The users goal is to [enter the phone number]. The user must [scroll to and select] the phone icon. The user can enter the number and then [press] the OK Button. Goal is now to [save contact]. User must [scroll to and select] Save Contact. User [selects Ok] and the goal is accomplished.

Problem Analysis

Problem N3f is a Responsiveness (feedback) issue, since the user must rely on a phone icon only, to indicate where the user must enter a number. It is also a Directness issue since there is a lack of visual cues.
Problem N3g: Is a Visual Appearance and Layout, Directness, Familiarity, Consistency and Predictability problem all rolled into one. In this instance the SAVE command is concealed on page 2 and must be scrolled to. This is a layout problem because this key goal (demonstrated by its presence in the NGOMSL sequence) is hidden away. It is inconsistent with other save options in the phone (see the multimedia tasks for instance). It is unfamiliar to the NGOMSL sequence (scrolling is never considered in the add contact method or its sub-goals or operators). Directness is involves, due to the fact this option is neither directly visible nor manipulated.

Figure 6-17: Add Contact to Screen for Phone-Contacts: Part 2

Problem N3h: Is a Directness issue, since no visual cue indicates the use of the enter key after the number is entered.
6.3.4 Task 4: Make call

In this task the users are instructed make a call from numbers stored in the phones call history which stores ten previously used numbers of two important different types: calls made, and calls received. To get to either, the user must either use a left or right key short cut to Calls Received and Right to Calls Made (Figure 6-18) or use the main menu instead. The Formatted Description for this task is as follows.

From the Start Screen.
The users goal is to make a call from call history. The first goal is to access the Call Records. User must decide if the Number has been dialled, or the Number has been received. If the goal is to go to calls made, the user may [press] the left Nav Key or go to calls made through the menu. If the goal is to go to calls received, the user may press the right Nav Key or go to calls made through the menu.

Figure 6-19: GSE Goal 4: – Make Call from Call History Part 1
Problem Analysis
In the first part of making a call from the call history, the user may either use the menu, or short cut keys. There is only one potential problem here.

Problem N4a: Is a Familiarity and Directness issue. The NGOMSL goals do not indicate that the user should go to USER DATA to find Calls made or received. Apart from this, it is difficult to determine any other major problems, in either part of the process.

In the second part of this task the user must find the phone number and start the call. Once the actual call is made, the sequence is the same as that for making a standard call in Task 2. The formatted sequence for part 2 of this task is as follows:

Formatted Description for Part 2
The Menu screen for User Data. If the user wants to go to calls made, they must select Option 4 Calls Made. This sets the calls list to calls made. If they want to go to Calls Received they must select Option 3 Calls Received. This sets the call list to calls received. If the call list is [empty] a warning appears saying no call in Calls Made or Calls received. If the call list is [not empty] the users goal is now to [search entries for the number they wish to call]. If the user decides the number is not here, they can return to the start screen. If the current number is correct the user can make a call, by [pressing the Send Button]. The user may look at the previous number by [pressing] the Nav Left Button. This sets the current record in calls made to the [previous one]. The user may also decide to look at the Next record, by [pressing] the nav right button. This sets the current record to the [next record].

An important assumption in this model, is that the when the end of the task records is reached it will cycle back to the start again. This applies in both directions. This task is straightforward and consistent with the NGOMSL model for this task. No obvious attributes are broken here.
Figure 6-20: GSE Goal 4: Make Call From Call History Part 2

Task 4, again demonstrates how a relatively short sequence of user goals is quickly inflated into a large GSE model once all system behaviour is added. Additionally, no low level NGOMSL operators are referenced here, although pressing the buttons, can be broken down into a locate, move hand and press sequence if necessary.

**Problem N4b:** Is a Directness issue, given that no visual cue is provided as to how the user can actually return to the start screen.
6.3.5 Task 5: Access 3 Web Services

In this task, users only had to press the 3 services key, to access web services. Very few user actions are needed. There is no other way of accessing the web services since it is not accessible from the menu or from short cuts.

The translated sequence is as follows:

**Formatted Description**
From the [Start] Screen. The user's goal is to [Access Web Services]. The first goal is to [connect to the web]. The user must [press] the Three Web Services Button. If another application in the phone is [open], the user can close the application and continue. Otherwise the user is still at the [start] screen. If the user [closes] the application or no other applications are [open], the [Connecting] Screen Opens. If the phone [connects] the screen changes to [web services], with an HTML [web menu] displayed and the users goal is [accomplished]. If the phone [can't connect] an error message appears, and [five seconds later] the user is returned to the [start] screen.
Problem Analysis

**Problem N5a** is a Flexibility issue. There is no other means of accessing the web. There is only one way to achieve this goal. This is also a potential Directness problem, if no visual way of accessing it or cues are provided.

**Problem N5b** is a Transparency issue. The user may be interrupted from their goals (connect to web) if another application is open. This is related to the multitasking features of the phone. It is also a Responsiveness issue, since the phone does not state which application is open. Directness and Predictability also apply.

**Problem N5c** is a Transparency, Responsiveness, and Error Recovery issue. The phone connection frequently disconnects breaking the NGOMSL task sequence, but the phone offers no reason as for why this might be the case, nor any support.

**Problem N5d** is a Visual Appearance and Layout issue, Responsiveness and also a Directness issue. This screen is very similar to other screens in its GSE representation. It is not immediately apparent if a visual distinction is made here.

**Important Note on Problem N5b**

Problem N5b is a critical and complex one. It reoccurs any time a user wishes to open an application within the phone. There are three types of applications each with limits on how many tasks can be open at a single time. When this problem occurs (and it does frequently), the error message contains no information on which task is actually open, or a direct means to get to that task. For the remainder of the study this problem will not be modelled in its entirety, but marked into the analysis section to demonstrate that this task is subject to this error.

6.3.6 Task 6: Navigate Web services to find Java game: Play Golf and download it.

In this task the subjects had to access the games screen from the web services, locate the game called “Play Golf” choose it and then download it to the phone. Once again the enter key is used to select the game. The task description begins in the manual at the start screen. However since this is already

![Figure 6.23: Choosing Games](image_url)
modelled in task 5, the task will be modelled from the web services screen.

The formatted description for this task is as follows:
From the [Web Menu] Screen. The User’s goal is to find the games site. User must navigate to the web icon games. The user’s next goal is to access the site, which the user does by pressing the Enter Key. This gives the user access to the [Web] Screen, containing HTML page [Games]. The next goal is to [choose a game]. The user must [scroll to and select an index] (corresponding to their game of choice). This changes the screen to the [HTML Games Description] To download the game, the user must [scroll to and select download game] The phone will either [begin downloading] or the [connection will drop out] If it drops out, the user is returned to the start. If the phone can [download the game], the Gamefile is downloaded from the network to the Java Applications. A message appears [confirming the download] and the goal is now [accomplished].

Predictions
Like the Web services, the Games functions are subject to connection dropouts. A potential problem is that the download game function is potentially hidden away below the instruction. A second smaller problem is that games cannot be downloaded directly from the games menu screen, and logically impacting on Directness. Also, no information, other than the flashing icon, demonstrates the download. Thus Responsiveness, Directness, Simplicity, Visual Appearance and Layout, are all potential issues.

Problem N6a: Is a potential Simplicity, Visual Appearance and Layout issue based on the number of icons on a mobile phone sized screen.
Problem N6b: Is another Transparency, Error Recovery and Responsiveness issue due to connection problems and poor feedback.
Problem N6c: Is also a Transparency problem, as well as an Error Recovery issue, owing to the fact that a file is also being downloaded, and no error support is shown in the diagram.
Problem N6d: Is really two problems. First it is a Predictability issue, as the game is saved to a Java Applications folder, and a Responsiveness issue, since no response as to where it has been saved is provided. Secondly the data transfer is assumed to be instant here, but there is no feedback until after the process. This is a potential Responsiveness problem.
Problem N6e is a potential Controllability and Forgiveness issue where external output (from the network is passed to the device but no way to cancel this is evident.
6.3.7 Task 7: Take 12 sec video sequence and cancel save

This task employs one of the most sophisticated functions available on any ubiquitous phone at the time of the study. Though many models offered photo cameras, few were able to take video sequences. The aim of this task was to employ the NEC’s Video Recording capabilities to take a short 12-second video. The 12-second duration is the maximum amount of footage that can be taken and played on other media enhanced phones, due to file restrictions concerning the media format itself. This is not like a video call, where the camera sends the images in real time to another phone sharing the same capability. This task was relatively simple. Multimedia is a menu option in its own right, and video is one of three media types (the other two being audio and still (photo) images. There is no need to stop the recording, in this task, since it stops at 12 seconds automatically. This condition is specified in the actual manual reference below, alongside the relevant NGOMSI model.

From the [Start] Screen

The users goal is to [take a video], and must [access the video camera]. The user [presses] the menu button, which opens the [main menu] screen. The user then [scrolls to and selects multimedia]. This opens the [Multimedia Options] screen. User [scrolls to and selects] Option 1: Video. This opens the [Video Options] Screen. The users goal is to [start recording]. The user must then [press] the enter key, which changes the screen to the [Video active] mode. If the user [presses stop], or [12 seconds elapse], the phone will [stop video recording]. The Phone will output the file to the collection of Video files. A message will be displayed, labelled [Make Video] informing the user with a message that [the file is saved].

Problems

Problem N7a: There is no visual cue as to the state of the enter button (Directness).

In NGOMSI the user must press the “Record Button” but the Enter Key is not marked so.

Problem N7b: The main problem with this task is that the file is saved automatically. This is a Controllability issue. In the actual user task, they had to cancel the “save” but this cannot be done here.
6.3.8 Task 8: Use calculator add (5 + 2)

This was the first of the tasks, assigned as an application. The calculator function is available on many phones, including both used in the study. The task was to enter a simple addition equation, get the result and then exit the calculator. Unlike the other phone, the NEC provided an image corresponding to the buttons on the screen. This is a challenge to a GSE representation since the image itself, contains important and meaningful information on the actual button layout.

Formatted Description – Calculator Part 1

The process for this task was modelled into two steps, in accordance with the NGOMSL goal sequence and the AS-IS instructions:
The two NGOMSI methods that apply are as follows

Method for Goal Use Calculator
1. Accomplish Goal: Access calculator
2. Accomplish Goal: Enter an equation
3. Decide if no more equations
4. Return with goal accomplished
5. Go to 2.

Method for goal Enter an Equation
1. Enter number
2. Enter operator
3. Enter number
4. Decide equation not finished go to 3
5. Get Answer
6. Return with goal accomplished

Formatted Description for Figure 6-27
The first part of the calculator model corresponds to the access calculator goal. The
Formatted Description for this is as follows:

From the [menu] screen, the user’s goal is to [access the calculator], the user [must [scroll to and
select] the Accessories icon which activates the [Accessories menu] Screen. The user [must scroll to
and select] Option 6 Calculator, which activates the [Calculator] Screen.

Figure 6-27: GSE Goal 8: Part 1 Accessing the Calculator

Problem Analysis

Global Problems: Problem N5b the multi-tasking problem applies to thus task.

Problem N8a: Contains the button layout for the keys but this will be dependant on
the user associating the picture and the keys with the functionality of the calculator.
No other supporting text is provided, so this is a potential Layout, and Simplicity
issue.
Node 3.7. in Figure 6-27 corresponds to the screen in Figure 6-28.

The second part of the Formatted Description corresponds to the NGOMSL method for entering an equation. This is as follows:

Formatted Description for Using the Calculator Part 2.
Goal Enter a Number. User Enters a Number and this number is added to the digits on the screen. User enters an operator, and if no value is stored, the digits are stored in memory and the operator is stored, the digits on screen are now [clear].

If the stored digits are [not empty], the calculator will [calculate the equation] which the calculator does by [calculating an answer] using the stored digits, the operator, and the digits on the screen. It will output this answer to the stored digits, and the screen digits.

If the user wishes to [start the equation again] the user can [press] the clear button, this stored digits to [empty], the screen digits to [empty], the stored operator to [empty]. If the user has [received an answer] the goal is accomplished.

Problem Analysis
In the second part of the model in Figure 6-29, there is another potential problem.
Problem N8b: Hitting the clear key clears all the digits. There is no undo, so this is a possible Error Recovery issue.
Entering the Operator is the third part of the GSE model. The obvious problem is whether the user is able or not to identify the buttons which correspond to the specific operators. The Formatted Description is as follows:

**Formatted Description for Using the Calculator – Entering an Operator**

**Goal [Enter Operator]**
User Decides to multiply, user must find and press icon number 2
  **Operator** is set to [multiply]
User decide to divide, user must find and press icon 3.
  **Operator** is then set to [divide]
User decide to add, user must find and press icon 4.
  **Operator** is then set to [add]
User decide to subtract, user must find and press icon 4
  **Operator** is then set to [subtract]
User decide to enter equals, user must find and press icon 5
  **Operator** is then set to [equals]

**Problem Analysis.**

**Problem N8c** applies to any operator decided upon by the user. This is a Predictability and Directness issue. Can the user make a connection between the multipliers and the correct icons? This is an issue better covered by a more detailed NGOMSL model.
6.3.9 Task 9: Take photo of glass and save

This task was extremely similar to task 7 where a video sequence was taken. In fact the entire procedure was almost identical, except that in this instance a single photographic image (and not a video sequence) is taken. The **Formatted Description** for this task is as follows:

---

**Formatted Description**

**Goal** [Take Photo] -- **Sub Goal** [Access Camera]

From the [start] **screen**, the **user** must [press] the **MENU Button**. This opens the [Main Menu] **Screen**. The **user** must then [scroll to and select] the **multimedia icon**. This opens the [Multimedia Menu] **Screen**. **User** must [scroll to and select] **Option 2 Image**. This opens **Screen** [Menu Images]. **User** must [scroll to and select] **Option 1 Take Photo**. This opens up **Screen** [Camera]

**Sub Goal** [Take Photo]

**User** must [press] the **Enter Key**, if the **filename** [saves], a **message** appears informing the **user** that the [filename] has saved. The **user’s goal** is now [accomplished], and the **user can** [Select] **Ok**, taking the user back to the [menu] **screen**.
Problem Analysis

Global Problems: This Task is vulnerable to the multi-tasking conflict.

Problem N9a: This task saves the photo automatically, satisfying a user goal automatically. However, the user does not get to decide to save, and so this is a Controllability issue.

Problem N9b: (not marked) Problems include a long task sequence and no shortcut. This is a Flexibility and Efficiency issue.
6.3.10 Task 10: Create appointment in calendar

In this task the user had to set an appointment, by accessing the calendar screen.

The Formatted Description for this task is as follows:

**Formatted Description for Create Appointment Part 1**

- **User's goal** is to [create appointment].
- **Users first Goal** is to [access calendar]. To do this the User must [press] the MENU button opening the [Main menu] Screen. The User must then [scroll to and Select] the Accessories Icon. This opens the [Accessories Menu] Screen. The User must then [go to] calendar by [scrolling to and selecting] calendar. The Phone must [check the calendar settings].
If the calendar view is [by week] the Screen displays the Calendar [Days in the week] If the calendar view is [by month], then the Calendar [days in month] is displayed. The user's next goal is to [go to the Date] The user must decide if the selected date is the desired date. If they do, the user must [navigate] to the date. The user may also [change] the navigation style. If the user decides the selected date [is the] desired date, the user must create an appointment, so the user must [select] create.

**Problem Analysis**
The Formatted Description can be found in Figure 6-33. The main problem in the above task was choosing between the calendar views.

**Global Problem:** The multi tasking problem encountered in previous tasks applies here.

**Problem N10a:** corresponds to the selection method between Week and Month views that demonstrates poor Consistency and Directness when switching to the day view.

**Problem N10b:** This problem is not marked, because it is manifested across Part 1 and 2 of Setting a reminder. Actually executing this process is extremely complicated when compared against the NGOMSL sequence. This is a Simplicity and Efficiency issue.

**Formatted Description Part 2:**

From the Menu Screen [create], If the user's goal is [save appointment] and Option 6 [is enabled], the user can [select] option 6 end, the appointment is then [saved] and the goal is [accomplished]

If the user's goal is to enter the details, the user must select option 4 details. This opens the text screen appointments. User then enters details into text field. When the user [is finished] the user must [press the Enter Key]. The details are outputted from the screen to the appointment and Screen [appointments] appears. User can then [select an icon]. After the user [presses] the Enter Key, the Appointment Icon [is added] to the appointment. If the appointment details are [not empty], Option 6 [end is enabled] and the details text is added to the appointment. If the Appointment details are [empty]. Option 6 remains [disabled], and user returns to the Menu Screen [create]

**Problem Analysis for Part 2**

**Problem N10c:** Is a Directness issue. The user must provide an icon, and there is no way to avoid this.
Problem N10d: The option to exit here is disabled automatically. This is a Controllability, Predictability and Directness issue.

Creating an appointment yielded four problems. Several were not obvious unless the model was checked against an NGOMSL goal sequence.
6.3.11 GSE Task 11: Create and send email

In this task, the users had to create and send an email, by navigating through the web services to the email services. Despite the connection between emails and messages, they are not accessible through the standard messages menus, which apply only to offline SMS and media messages. An additional complication is that the email services do not appear on the main web services screen, but are accessible through the more - hyperlink and not as directly accessible as games.

The relationship between the Web Services Screen and More Services can be found in Figure 6-36. The Formatted Description of this task is as follows.

**Formatted Description**

After accessing the [Web Services] Screen, if phone [cannot connect], a connection error displays, system waits [five seconds] and return to start screen.

If phone [can connect], [Web Services] Screen appears with HTML [Web menu].

User's [goal] is to [find email site], the user Must [navigate to] the web Icon More and [press] enter key. Screen [web] opens up with HTML [more].

User must then [navigate to and click] LINK [messaging]. This opens up Screen [web] with HTML [Email Home]. The user's goal is to [create an email], so
[they scroll to and select] **create email**  This opens up **screen** [web] with **Page** [create email]

---

**Problem Analysis for Task 11 in Figure 6-37.**

**Global Problems:** Because Email is web based it is subject to the web connectivity error.
**Problem N11a:** (not modelled in above diagram) is a potential Directness error relating to choosing web services to send email.

There are two problems in Figure 6-37.

**Problem N11b:** This is a Directness and Efficiency issue. Email is not immediately accessible from the web services, and the user must go through the more Internet option. Consistency and Familiarity are also involved.

**Problem N11c:** This is a Consistency issue. The option "messages" matches the term messaging where SMS messages are entered and used. However, in this instance it only applies to email.

![Main Email Screen and Create Email](image)

**Formatted Description – Send an Email Part 2 in Figure 6-39**

The second part of the task concerns creating the actual Email. The **Formatted Description** for this task is as follows:

**User** [press] **Nav Left**, opens **Screen** [web] with **Html** [home]

**Goal** [Enter Address]. **User** must [Scroll to] **Text Field** [address] and [press] **enter**

This opens **Text Screen** [edit]. The **user** must then [enter text], the **user** must press [enter], the **text field** outputs its **details** to the **Email Message**. Returning the **user** to **screen** [connected] with **HTML**. **Goal** [Enter Subject]. The **user** must [scroll to] **text field** [subject] and [press] **Enter**. **Email** outputs the **subject** to **Text Screen** [edit]. The **user** [goes back] is [enter message], the **user** must [scroll to] the **text field** [messages] and [press] **enter**. **Email** outputs its **message** to the **text screen** [edit]. If the **goal** is to [send a message], the **user** must [scroll to] **link** [send message] and [press] **enter**. **Phone** sends **message**. If the **goal** is [accomplished] the **user** is sent to
Screen [connected] HTML [mail information] Otherwise the [Mail Information] Page is opened, with a message that [the email is not set]. If the user [wants to return to create email] The user must [press ] NAV left.

Problem Analysis

Problem N11d: Is a Directness issue; the user has to open the text fields on screen before entering text by pressing the enter key. The text cannot be entered directly into the text fields (see Create Email screen in Figure 6-39).

Problem N11e: Is a similar situation. This is also a Directness issue, since the text cannot be directly entered. The user must select the text field, press enter, and wait for an additional text input screen to appear. This does however exhibit Consistency with other instances using the text entry key.

Figure 6-39: Creating and Sending an Email Part 2
6.3.12 Task 12: Create SMS message with “Hello” - Save it to memory

Unlike the previous task creating an SMS message requires no web access at all. Instead, the user may opt to access the message functions through the menu, or alternately via the left selection key.

Formatted Description – Part 1

**Goal Access Messages**

From the start screen User must [decide to select messages] Or [decide to use menu]
The user must [press] the menu button. After the menu screen appears the user must [select ICON messages]. This opens Menu screen [messages]. The users goal is to [create a message] The user [selects] option 1 create message. This opens Menu Screen [Message Type]. The user must then [select create text message].

Problem Analysis

There are no obvious problems in the first part of this task. The Goal structure in the model is preserved, and short cuts are available.
Formatted Description Create an SMS Message Part 2

The behaviour in Figure 6-41 concerns the actual creation of the message. The translated text is as follows:

From **screen** [create messages]
**User** may [decide to exit] or [achieve goal send message], [enter message] or [enter address]

If the **user** [decides to exit], dialogue [End Editing] appears with the **message** "do you want to exit without saving the changes." If the **user** [selects yes], they are returned to **menu screen** [message]. If the **User** selects no they go to **Screen** [new text]

If the **goal** is to [send a message]
The **user** must [select Send]. If the **address** is [not empty], the **phone** will [send] **message**, confirming this with a **process dialog**. If the **phone** [detects the message] has been sent, a **dialogue** appears with the [recipient details]. The **goal** is [accomplished] and the **user** [may select] **OK** to finish.
However if the message [failed], a [can’t send] error appears, and the user must select Ok to [return to] the screen [Create message]

If the goal is to [enter a message], The user must [highlight and select] the text field message. This opens text screen [edit message]. When the user [is finished entering the message] the user must [press] enter, the text field message is now part of the SMS message.

If the goal is to [enter an address], The user must [highlight and select] the text field 2 address. This opens text screen [edit message]. When the user [is finished entering] the message the user must [press enter], the text field message is now part of the SMS message.

Problem Analysis

In the second part of the task, the user creates the actual message. There are several potential problems here.

**Problem N12a:** This is a Directness issue. The text fields cannot be edited directly, they have to be opened, by clicking on them and loading up an extra screen. The messages screen can be found in Figure 6-42.

**Problem N12b:** This is a Forgiveness issue. The user is able to send a message with an empty message, and an empty address.

**Problem N12c:** This problem recurs several times. If a message fails to send, the user is sent straight back to the create message screen to start again. This is an Error Recovery issue.

This task demonstrated the NEC’s response to invalid text data. The NEC will not warn a user that a text field has been left blank, or seek to address it. Keeping in mind that trade-offs are a big part of design, this may well aid the Controllability of the product, by freeing the user of restrictions. Whether this proves to be an issue will be born out in the usability tests.
6.3.13 Task 13: Delete appointment in calendar

The latter tasks in the user study are designed to undo or use, the results of earlier tasks. In this task the user has to delete the appointment they created in Task 10. There are several ways of deleting the appointment. The User can delete all the appointments or a single appointment. Users will be free to choose either.

<table>
<thead>
<tr>
<th>Calendar [week view]</th>
<th>Instructions [extract]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Press the Menu key</td>
<td>• Scroll to Accessories and Select</td>
</tr>
<tr>
<td>• Scroll to the appointment date and press the Enter key</td>
<td></td>
</tr>
<tr>
<td>• Scroll to the appointment and Press Submenu</td>
<td></td>
</tr>
</tbody>
</table>

### Figure 6.43: GSE Task 13: – Deleting an Appointment

The Formatted Description for this task is as follows:

From Screen [Calendar View]
Decide Selected Date is not desired Date
The user can [navigate to] desired date or [change navigation Style]
If the user [decides the selected date is the desired date], the user can [decide to Delete all appointments] or [Delete a Specific Appointment]

If user [decides to delete all], the user must [select] Submenu, which displays the calendar submenu. User can [delete all], by [selecting] Option 5 delete all, or they can [delete all up to Yesterday] by [selecting] Option 4 delete up to yesterday. Or they [select] delete all, a warning dialog with [Delete All] is displayed. If user [selects yes], the phone [deletes] all the appointments, leaving all appointments [deleted] and displays a message [Appointment deleted] and the goal is [accomplished]. If they select [no] they return to screen [calendar view].

If the user [deletes a specific date], the user must [press] Enter. This displays the [Day view] screen. The user must [select] the submenu, to display the Day submenu. User must select Option 5 Delete, a warning dialog appears with message [delete?] if user [selects] no they return to the [calendar] screen. If they choose yes, the phone [deletes the appointment], the specific appointment is [deleted] a message says [appointment has been deleted]. The goal is [accomplished] and the screen goes to [day view].
Figure 6-44: Deleting an Appointment

Problem Analysis

**N13a:** Is a Consistency and Simplicity issue. There are multiple ways of navigating depending upon the current view. To change this requires going through a Sub-Menu.

**N13b:** This is a Directness and Predictability Issue. To delete a specific appointment the user has to go to the Day View. It cannot be done from Weekly or Monthly views.
6.3.14 Task 14: Send stored photo to 0412 3848 10

The manual instructions aren’t completely compatible with this task. The manual instructions contain a procedure for sending a freshly taken image. However in this task, the users were expected to find the photo they saved in Task 9 and send it to the same number used through out these tests.

Formatted Description Part 1: Figure 6-45

The Formatted Description for Part 1 of this task is as follows: From Screen [Start] The first goal is to [Access Camera] User must [Press] the Menu Button to open Screen [Main Menu] [Scroll to and select] multimedia icon to Open Multimedia Menu Screen. [Scroll to and select] Option 2 to Open Screen [Images]. [Select] Option 2 Viewer to Open Screen [Image Viewer]. The user may [view image], by [pressing Selecting] Display this opens Screen [Display] User [may quit], which returns them to screen [image viewer]

The users second goal is to [select photo]. User [scroll to and Select Photo]. User must [select Submenu]. To Open screen [submenu], user must [select] option 1 Send image

Problem Analysis

N14a: Violates Directness and Efficiency. The user cannot view the photo, and send it directly, they have to quit the viewer first.
N14g: Violates Directness and Familiarity. To send a saved photo message the first goal is to access the camera.

Part 2 of the Send Stored Photo Process Figure 6-46

In the second part of the send stored photo task, the user has to actually enter the message.

Formatted Description:

From screen [composer] the user can [choose to preview the picture] by [selecting] preview, opening up a [display] screen of the PHOTO.

The user may [choose to accept] the photo by [pressing] Enter. This opens the [new messages] screen. The users goal is now to [send] the message. The user must [select] Send. If the address [is empty], and error message displays, telling the user “Cold not send - Check Recipient Details” The user [selects] Ok and returns to the [create message] screen

If the address is [not empty] the Phone must [send] the message. This opens a dialog, showing the message being [sent]. If the message is [sent] a dialog opens
confirming that the messages [have been sent]. The goal is [accomplished], and the user must press ok to [finish the task].

If the goal is [Choose Recipient], the user must highlight and [select] the address text field. This [opens] text screen [edit address]. When the user is [finished editing] the address (which is numeric) the user must [press] enter. The Text field outputs its address to the SMS address and the screen returns to the [new messages] screen.

The user may [choose another] photo, by [selecting] the cancel option. This opens an “end editing” dialogue. If the user [decides to exit], they [select] “Yes” and are returned to the [start] screen.

If the user selects [no], they are returned to the [composer] screen.

---

**Figure 6-45: NEC Task 14: Send Stored Photo – Part 1**

Page 296
Figure 6-46: Task 14: Send Stored Photo Part 2

Problem Analysis.
Part 2 of the task yielded a large number of potential problems

**N14b:** There is no visual cue for pressing Enter as a means of selecting the photo moving on. This is a Directness issue.

**N14c:** The Preview window previews the photo, much like Composer screen. But there is no option to move on from here. And functions are inconsistent with those of the composer screen. This violates Consistency and Directness.
N14d: There is a possible mismatch here. An SMS message is sent to an address, which can only be entered as a phone number. It is not possible to enter a name here. This is a Predictability problem.

N14e: When a message fails, the phone returns to the start screen. This is an Error recovery and Efficiency error.

N14f: To choose another photo, the whole message must be scrapped. This is an Error Recovery, Directness and Efficiency error.

N14g is a Directness and Efficiency error relating to accessing the camera.

Summary of Task 14
Task 14, has yielded a large number of potential problems, some more obscure then others. GSE screen notation makes it possible to see that viewing the image is spread across multiple screens that have different options.

6.3.15 Task 15: Activate java game: Begin game: Close java game

In this task the users are expected, to find and activate the game they downloaded in Task 6. Provided the application is stored on the phone, the users are able to complete this task.

Formatted Description:

From the [main menu] screen User [Scroll to and select] Java Icon, opening Menu Screen [Java]. User must then [scroll to and select] applications. Their goal now is to [choose a game], the user must then [scroll to and select] an application. This opens a screen for [describing game]. User can then choose [select]. A splash-screen for [Java] appears. The user has the option to quit. The phone [starts the] application, and the application is now [started]. Opening up an [application] screen.

Figure 6-47: GSE Task 15: Applications Screen.
Problem Analysis

N15a: This is a Predictability and Directness issue. This violates the NGOMSL model for this method. Why are games stored under Java?

N15b: This is a Predictability and Responsiveness issue. Games are stored under Applications. The term “Games” is never used in the menu.

N15c: This is the same error as N15b.
6.3.16 Task 16: Delete from phonebook

In this task, the users have to delete all the records, matching "Al" from the phonebook. The users have the option to delete all the records, or just those specifically marked "Al."

Part 1: Figure 6-49

In part 1 the user accesses the contacts and then chooses between deleting one and deleting them all.

From the [Contacts] Menu Screen

Goal [Access Contacts] User must [select] contact. This opens Menu Screen [contacts menu]. User may choose to [Delete all] or [Delete a specific contact]. If the user decides to delete all contacts the user must [select] option 6. After the Dialog [Delete Screen] appears if they select yes, [confirmation] message appears, and the goal is [accomplished]. If they [select] no they are returned to the [Contacts menu screen].
**Problems:** No obvious problems are apparent in part 1 of this task.

**Delete Contacts Part 2 in Figure 6-50**

In this task the user searches for contacts named “Al”.

The **goal** is to [delete a specific contact]. To do this the **User** must [find the contacts] by [selecting] **Option 1**. This opens the [Search Contacts] **Dialog**. **User** can [decide to show all contacts] by [pressing] **option 1** or [list specific contacts] by selecting **option 2**. If they press 2, the **phone** will automatically check the **contacts** list.

---

If the user [presses] option 1, the **text screen** [name input] is displayed. If the **user** [accepts the name] the **user** can [press] the **enter key** to accept it. The **contacts** list is now set to all those whose [name is Al] and the **Phone** now [checks the list for these contacts].
Delete Contacts: Part 3 in Figure 6-51

In the final part of the Delete Contacts Procedure, the user must select the actual record to delete it. The Formatted Description is as follows:

**Formatted Description:**

If the number of contacts is not [greater than 0] the goal is [accomplished], a message displays saying [no records have been found], the user [presses] Ok and is returned to Menu Screen [search contacts].

If the number of contacts is greater than 0. The Menu Screen for [Contacts List] opens. If the user [verifies that all target contacts are deleted], the goal is [accomplished]. The user may [select] a contact on the screen. This opens Dialog [Delete]. If the user [selects] Yes, a [delete] confirmation appears, the phone [pauses two seconds] and returns to the menu screen [contacts list].

**Problem Analysis**

**Problem N16a:** The goal is to delete specific contacts, but only one may be deleted at a time, even if multiple records exist. (The only alternative is to delete all) this is an Efficiency error.
6.3.17 Task 17: Delete stored photo

Deleting a stored photo is accomplished through the image viewer. This task is not analysed or modelled in significant detail.

6.3.18 Task 18: Delete SMS message

Deleting an SMS message is accomplished from the menu system used in Task 12. This task is not analysed or modelled in significant detail.

6.3.19 Task 19: Turn phone off

The final NEC task, to turn off the phone is almost identical to the procedure for turning it on, including yielding the same problem.

Problem Analysis

N19a: there is no acknowledgement the phone is turning off until it begins. This is a Responsiveness issue. The time delay is an Efficiency and Directness issue.

N19b: There is no warning dialog if an application is open and data will be lost. This is a Responsiveness and Forgiveness issue, as well as a Controllability issue since the procedure cannot be cancelled once it begins.
6.3.20 NEC Phone Summary

<table>
<thead>
<tr>
<th></th>
<th>Task</th>
<th>Problems</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Turn Phone On</td>
<td>1</td>
<td>EF LN RS DR</td>
</tr>
<tr>
<td>2</td>
<td>Phone Call</td>
<td>5</td>
<td>RS RS RS EF SM CS</td>
</tr>
<tr>
<td>3</td>
<td>Add Contact</td>
<td>8</td>
<td>PR SM DR FM TN FG TN PR CS RS DR VA DR FM CS PR DR</td>
</tr>
<tr>
<td>4</td>
<td>Call History</td>
<td>2</td>
<td>FM DR DR</td>
</tr>
<tr>
<td>5</td>
<td>Access Web</td>
<td>4</td>
<td>FL DR TN RS DR PR TN RS ER VA RS DR</td>
</tr>
<tr>
<td>6</td>
<td>Download Game</td>
<td>5</td>
<td>SM VA TN ER RS TN ER PR RS CT FG</td>
</tr>
<tr>
<td>7</td>
<td>Take Video</td>
<td>2</td>
<td>DR CT</td>
</tr>
<tr>
<td>8</td>
<td>Calculator</td>
<td>3</td>
<td>VA SM ER PR DR</td>
</tr>
<tr>
<td>9</td>
<td>Take Photo</td>
<td>2</td>
<td>CT FL EF</td>
</tr>
<tr>
<td>10</td>
<td>Appointment</td>
<td>4</td>
<td>CS DR SM EF DR CT DR PR</td>
</tr>
<tr>
<td>11</td>
<td>Email</td>
<td>5</td>
<td>DR FM PR DR EF CS FM CS DR DR CS</td>
</tr>
<tr>
<td>12</td>
<td>Create SMS</td>
<td>3</td>
<td>DR FG ER</td>
</tr>
<tr>
<td>13</td>
<td>Delete Appointment</td>
<td>2</td>
<td>CS SM DR PR</td>
</tr>
<tr>
<td>14</td>
<td>Send Photo</td>
<td>7</td>
<td>DR EF DR FM DR CS DR PR ER PR ER DR EF</td>
</tr>
<tr>
<td>15</td>
<td>Start Game</td>
<td>3</td>
<td>PR DR PR RS PR RS</td>
</tr>
<tr>
<td>16</td>
<td>Delete Contacts</td>
<td>1</td>
<td>EF</td>
</tr>
<tr>
<td>17</td>
<td>Delete Photo</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Delete SMS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Turn Phone off</td>
<td>2</td>
<td>RS EF DR FG CT</td>
</tr>
</tbody>
</table>

Table 6-2: NEC potential problems summary

The NEC GSE models in this section were built from formatted task descriptions, based on the user’s goals for the tasks, but primarily on the descriptions of the tasks provided by the manuals. These task models for the NEC tasks predicted 51 problems (see Table 6-2) across the 19 different tasks (corresponding to the tasks conducted in the user tests), when compared against the attribute representations in Chapter 4. The task that yielded the most problems was Task 3: adding a contact. This task was complicated by different data sources and a lack of visual cues. Other problematic tasks included sending a photo, sending an email and making a phone call. This concludes the NEC task modelling. The analysis of the GSE models demonstrates that a significant number of problems could be extracted from the GSE models. The next section, models the 16 tasks conducted on the other phone used in the main study (the Samsung T100), and analyses these models to find more potential usability problems. These problems are then compared against the test findings in the next chapter, to see just how effective using NGOMSL based GSE models to find usability problems actually is. However, the 51 NEC problems, have demonstrated that a number of significant design issues can be identified without actual users.
6.4 The Samsung GSE Model

The Samsung Phone is smaller, older, and has less buttons than those on the NEC. This phone does not use Java applications or an inbuilt camera, though it does offer web connectivity and voice recognition. This model was chosen on the basis of its availability, as well as its differences with the NEC model. A significant difference with this phone is that its main menu scrolls not only up and down, but moves left and right also. Unlike a computer main menu though, you cannot see the options to the left or right.

Figure 6-53 shows the button layout of the Samsung T1000. Figure 6-54 shows a standard menu screen on the T1000 and shows the idle screen, which is the starting screen for most activities.

<table>
<thead>
<tr>
<th>Idle Screen</th>
<th>Standard Menu Screen</th>
</tr>
</thead>
</table>
| YES OPTUS Saturday 12 Nov 18:17 | Menu Number Menu Title
| Menu Phonebook | Animated Backdrop (4 frames) |
| | Menu Position Option
| | Option Number Selected Option
| | Missed Calls Option
| | Received Calls Option
| | Dial 10 Entries Option
| | Left Soft Key Navigation Options Right Soft Key
| | Total Entries |

Figure 6-54: Samsung T100 Idle and Menu Screens
6.4.1 Tasks in User Tests

The task instances that are covered on this phone in the user tests are as follows:

1. Turn phone on
2. Make call to 0407 515 530
3. Add following names and number to phone book (contacts)
   Name: Al, Number: 0407 515 530) to phone book (contacts)
   Name: Al, Number: 3848 2885 to phone book (contacts)
4. Activate Call history – find 0407 515 530 and return call using call history
5. Access Web Browser
6. Use Speed Dial – Al (0407 515 530)
7. Create SMS: with “Hello” as message. Save message
8. Use calculator add (5 + 2)
9. Assign Voice Recording to Al (phone book record)
10. Create appointment in Scheduler for October 19 2004 2 30 pm lunch.
11. Use Voice Dial to call – [Al]
12. Delete appointment in Scheduler for October 19 2004: 230 pm
13. Send saved SMS “hello” to Al
14. Delete Al from phone Boo (contents)
15. Turn Phone off

The remainder of this section contains GSE models of these activities based on the NGOMSL models conceived in section 1. As in the NEC analysis, the Samsung models are analysed looking for usability problems based on the attributes modelled in chapter 4. Many of the tasks in the Samsung correspond to those on the NEC, since the user test was intended to be comparative. There are several physical issues, which are unable to be addressed. These include the size of the buttons that are incredibly small. Thus all the GSE models are based on the assumption that the user is able to press the buttons successfully.

Now that the tasks have been specified, it is time to analyse the Samsung model, beginning with the first task allocated to the users in the user tests; Turning the phone on.
6.4.2 Task 1: Turn phone on

In this task the user had to activate the phone. Unlike the task 1 on the NEC, the users had to enter a password to activate the phone. The **hold end key for one second**, is similar to the process in the NEC model, except that no timer component is used. The same problem can be identified though.

**Formatted Description** (Figure 6-55)

Same Preconditions apply as in NEC Task 1

**Goal** [Turn the Phone On]
User [Opens] the **Phone and the User** must then [Hold the end key for one second]. Phone may ask for password, opening the **password** [screen]. If the **password** is [accepted], the **Phone** [Registers with the network provider] as normal. The **phone** is now [on] and **Screen** [idle] opens up. **Goal** is now [accomplished].

![Figure 6-55: Samsung Turning the Phone On.](image-url)
Problem Analysis

Problem S1a: Is the same problem as N1a, and concerns the lack of response while holding the button. Thus this is a Responsiveness problem. The duration is shorter, so this limits this a problem.

Problem S1b: corresponds to entering the password. This is an Efficiency problem, however it is necessary for phone security. So this is a usability trade-off in favour of a non-usability attribute.

Turning the phone on demonstrates how trade-offs are an integral part of the design. This also demonstrates how the goals in a NGOMSL models, can be omitted if the users task requirements aren’t properly specified.

6.4.3 Task 2: Make call to 0407 515 530

In this task the users had to make a standard phone call by keying in each number manually. Part 1 of this task corresponds to entering the actual digits.

Formatted Description

Goal [Make a Phone Call] from Screen [idle]
Users goal is to [start call]. User must [decide to make a manual call]. User may then [enter] a digit, which becomes the first digit in the phone number. The screen [Enter Number] appears. The user can [decide to delete all digits] by [holding down] the clear key [for five seconds]. This [clears] the phone number. The user can [choose to delete a single digit] by [pressing] the clear key as normal. This removes the last digit from the phone number. The user can opt to enter a digit which is added to the end of the phone number.

When the user’s goal is [activate call], the user [must] press the send key. This sends the phone number from the screen to the phone, and the phone [starts connecting].

Problem Analysis

There are three potential problems in this model.

Problem S2a: Concerns the lack of visual cues, on entering an actual number. This is a Responsiveness issue.

Problem S2b: Corresponds to pressing the Send Key to activate the call. It doesn’t necessarily fit into the NGOMSL goal structure (a Directness and Familiarity issue).

Problem S2c: Concerns the dual nature of the clear button. This is a potential Forgiveness and Predictability issue, if the user holds the button for too long.
Formatted Description of Part 1

In part 2 the user actually conducts the phone call.

Form Screen [Dialling] If Call is [Busy] Phone Call [Stops, and [user busy] alert appears If the phone is in [Redialling model] the phone pauses then a [redialling] alert appears and the phone [redials again]. Otherwise the is returned to screen [idle].

If the phone [connects], the phone is [connected] the [call] screen appears, until the user [decides its time to end call], and may [select] end or [press] the end call key, a [call ended] alert appears the phone [pauses], and the screen returns to idle.
If the user [decides to abort the call], they [press] the end key and the call [is aborted], an [aborted call] alert appears the phone [pauses] and the screen returns to [idle] mode.

**Potential Problems – Make Call Part 2**

**Problem S2d:** The only clear potential problem in Figure 6-57 is that on a bust signal, if the phone is not in redialling mode, the user is returned to the idle screen, and the number must be re-entered. This is a potential Efficiency and Error Recovery problem.

That concludes the modelling for making a standard phone call. However tasks 4, 6, and 11 cover additional ways of calling the user.
6.4.4 Task 3: Add following names and number to phone book (contacts)

- Name: Al, Number: 0407 515 530) to phone book (contacts)
- Name: Al, Number: 3848 2885 to phone book (contacts)

In the third task the users were asked to enter two numbers (one mobile number and one local number) under a single name. This is the same as NEC task 3. However, unlike the NEC, the Samsung does not support multiple numbers for a single contact. Thus the user must enter two records, for each number, although the names can be the same across both. An additional difference with the NEC is that there is only one-way of entering a new contact/phone number, unlike the NEC which has at least two.

The other major difference is that the Samsung does not ask for a destination every time the user adds a contact as in the USIM/Phone Memory choices on the NEC. Figure 6-58 contains the four main screens involved in this process.

![Figure 6-58: Samsung Task 3: Adding a Contact](image)

**Formatted Description** (first part is the almost identical to Task 2 except for goals)

The Formatted Description for this task is as follows

**Goal** [Add a contact] from **Screen** [idle]

Users goal is to [enter number]. User may then [enter] a digit, which becomes the first digit in the **phone number**. The **screen** [Enter Number] appears. The **user** can decide to delete all digits by [holding down] the **clear key** for five seconds. This clears the **phone number**. The **user** can choose to delete a single digit by [pressing] the clear key as normal. This removes the last digit from the phone number. The **user** can opt to enter a digit which is added to the end of the phone number.

When the user's goal is [add to address book], User must [select] save, the **Screen** sends the **number** to memory. **Screen** [enter name] opens up when the user [decides the name is correct!] the user must select save, and the name is added to the new record. **Screen** [location] appears, User can [save record to current Destination], or
[change Destination]. If the user changes destination, they must select “To destination”. Then if the Destination is [the phone], it changes to [the card], and if the destination is [the card] it changes to [the phone]. When the user is ready to [save the record] to the destination they must select [ok], the phone [sends the] new record, [to the] Destination. And the goal is [complete]. Screen returns to [idle].

**Figure 6-59: Samsung Task 3 Add to Phone Book**

---

**Problem Analysis.**

In many ways this process is simpler than the NEC task. However it does have its share of problems.

**Problem S3a:** Same as S2a in Samsung Task 2

**Problem S3b:** Same as S2b in Samsung Task 2

**Problem S3c:** This is a Flexibility, Directness and Predictability issue: there is no other way of adding a contact to the phonebook, assuming that this is a complete representation of adding contacts.

**Problem S3d:** The screen automatically returns to the starting or idle screen, this is a Directness and Predictability issue.
That concludes adding contacts on the Samsung. There is a goal discrepancy, in the sense that accessing the contacts or phonebook is not part of the process. This is a Predictability and Familiarity issue.

6.4.5 Task 4: Activate call history – find and return call using call history

In this task the users were asked to dial a number from call history. Like on the NEC they had the option of using either calls received, or calls missed. The options and screens involved can be found in Figure 6-60.

![Screens](image)

**Call Records**
Via this menu, you can view the phone calls:
- Missed
- Received
- Dialed

The number and name (if available) are displayed together with the date and time at which the call was made. You can also view call times and costs.

**Missed Calls Menu 1-1**
- Lets you view the last 10 unanswered calls. You can also:
  - Edit the number if available and dial it or save it in the phonebook
  - Delete the number from the list

**Received Calls Menu 1-2**
- Lets you view the last 10 phone calls received. You can also:
  - Edit the number if available and dial it or save it in the phonebook
  - Delete the number from the list

**Figure 6-60: Samsung Task 9 Using Call Records.**

**Formatted Description: Figure 6-61**
The Formatted Description for this task is as follows:

**Goal** [Return Call] from Screen [idle]  
First goal is to [activate] the call history. User must press the [menu key]. This opens the Menu Screen [Call records].

**User** can [decide to use calls missed], by selecting option 4. The label is set to [calls missed], and the call list is [set to Calls missed]

**User** can [decide to use calls received] by [selecting] option 3, the label is set to [calls received] and the call list is set to Calls received

**User** can [decide to use calls dialled] by [selecting] option 2, the label is set to [calls dialled], and the call list is set to calls dialled.

Screen [Calls] is opened up and the User may choose to [make call] with the selected record. If the call list is [empty] they are returned to screen [Calls]. If the selected record is [valid] the phone [dials] the number and the goal is [accomplished].
Problem Analysis

There is no obvious problem here, other than the call list perhaps being empty. This task does not have a problem with the menu, since this menu screen is the first to appear when the menu button is pressed.

This concludes the second of the make call options. The next task is a brief look at the Web connectivity options on the Samsung.

6.4.6 Task 5: Accessing the Web Browser

Unlike the NEC task, this phone did not have direct access to a service provider. Instead this task was designed to see how users would choose to access the web.
functions. On this phone, unlike the NEC, several methods were available, including the unusual side-scrolling menu. The Samsung's web capabilities, and menu screen are demonstrated in Figure 6-62.

<table>
<thead>
<tr>
<th>WWW Menu Screen</th>
<th>Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>8. WWW Services</strong></td>
<td></td>
</tr>
<tr>
<td>![Satellite Dish Animation]</td>
<td></td>
</tr>
<tr>
<td>1. Home</td>
<td></td>
</tr>
<tr>
<td>2. Bookmark</td>
<td></td>
</tr>
<tr>
<td>3. Goto</td>
<td></td>
</tr>
<tr>
<td>Select</td>
<td><strong>WWW Services</strong></td>
</tr>
<tr>
<td>You can use your phone as a Web Browser.</td>
<td></td>
</tr>
<tr>
<td>You can obtain up-to-date news, weather, sport, and other information as well as use e-mail and other Internet services. Please contact your service provider to open your Internet account. To use the Web Browser, you must indicate your phone number via the Own Number menu option (5-1-2).</td>
<td></td>
</tr>
<tr>
<td><strong>Home Menu 8-1</strong></td>
<td></td>
</tr>
<tr>
<td>To launch the browser from the idle mode, press the key in the centre of the scroll button.</td>
<td></td>
</tr>
<tr>
<td>Once connected, the homepage is displayed.</td>
<td></td>
</tr>
<tr>
<td>The content depends on the service provider.</td>
<td></td>
</tr>
<tr>
<td>To exit the browser at any time, press the key.</td>
<td></td>
</tr>
<tr>
<td>The idle screen is displayed.</td>
<td></td>
</tr>
</tbody>
</table>

Figure 6-62: Samsung Task 5 Accessing the Web Browser

**Formatted Description:**

The translated behaviour for this task modelled in Figure 6-63 is as follows:

**Goal** [Access Web Services] from **Screen** [idle].
**User** may [decide to use Menu], or [decide to use key]. If **user** decides to use key and [presses] the **centre key**, the **phone** [starts connecting]
If the **user** [decides to use menu], the **user** must [press] the **menu key**. The **user** must then [navigate to WWW services]. **Menu screen** [web services open]s. **User** can [select] **Option 1**: **Home**. The **phone** [starts connecting]. If the **phone** [connects]... (not covered in task) if the **phone** [can't connect], **Message** appears with [please set up your own number], the **phone** [pauses a few seconds], and then the **message** [disappears].

Page 315
Figure 6-63: Task 5: Samsung Accessing the Web

Problem Analysis:

Problem S5a: Is a global problem for any task where the user to navigate through the menu. The side-scrolling menu, impacts on Visual appearance and Layout, Simplicity Directness, Consistency and Familiarity. Functionality is contained in a subsystem.

Problem S5b: Is a problem specific to this task, the message is an ambiguous response. This impacts on Responsiveness, Error Recovery, Simplicity and Transparency.
6.4.7 Task 6: Use Speed Dial

The third way of making a call covered in this study is to use Speed Dial. Speed dial is a way of activating pre-set phone numbers in the contacts list by holding down the corresponding speed dial number on the keypad. However Speed dial numbers can be in double digits, which requires multiple key presses.

The Formatted Description for this task is as follows:

**Goal [Speed Dial] from Screen [idle]**
User can either recall number or not. If the user cannot recall number, the user must [find Entry in the phone book] This will take them to the [Record] screen.

If the user [can recall number], user must [decide if number is between two and nine], or [ten or more].

If the number is between [two and nine], the user must [hold down] the corresponding numeric key, the Phone [retrieves the phone number] and [connects].

If the number is [10 or more] the user must [enter the numbers] then [press] the # key. The phone will [find the record], and open up screen [record]. If the user [decides to make call], they must select Dial, the phone starts [connecting] and the goal is [accomplished].
Problem Analysis:

Problem S6a. Speed dial is supposed to be a rapid alternative to dialling. However, if two or more digits are involved, the user must go through the record screen, as they would if they were looking for the number in the phone book. This is a major Efficiency, Consistency and Predictability issue.

Problem S6b. There is no way to check the speed dial numbers without going to the phonebook. This is a Directness, Efficiency, Learnability and Flexibility issue.

The above problem demonstrates how a GSE or NGOMSL analysis, can demonstrate unnecessary complexity in a task. The speed dial procedure continues on into the dialling procedure as documented in part 2 of task 2.
6.4.8 Task 7: Create SMS: Save message

In this task the users had to create a simple SMS message as they had to in NEC task 12. The important screens in this process can be found in Figure 6-66.

![Figure 6-66: Samsung Task 7 Create an SMS message](image)

**Formatted Description: Figure 6-67**

The Formatted Description for this task is as follows:

**Goal** [Create SMS] from **Screen** [idle] **User** [decides to use key] they can **press** the **nav left key** taking them to **menu screen**[messages] If they [decide to use Menu], the **user** must press **menu** key. **This opens up the menu screen** at [call records]. The **user** must [navigate] to **Menu Screen** [Messages]

From **menu screen** [messages] the **user** [selects option 2 to write messages]. **Text Screen** [enter name] then opens. If the **user** [decides the message is ok], the **user** should [select ok]. If the **message** is [empty] screen returns to **screen** [enter name] Otherwise the screen returns to **menu screen** [send message] **User** may [decide to save only]. The **message** is [saved]. This returns them to **menu screen** [read messages] and the **goal** is [accomplished].

**Problem Analysis:** No major problems evident
Figure 6-67: Task 7 Create SMS
6.4.9 Task 8: Use calculator add (5 + 2)

In task 8, the users had to enter the same equation (5+2) as in the NEC task 8. The calculator on the Samsung differs both in layout, and how the operators are entered. The two main screens involved in this task can be found in Figure 6-68.

![Figure 6-68: Samsung Task 8 Use Calculator](image)

**Formatted Description**

The Formatted Description for this task is quite detailed, but demonstrates the way the calculator screen operates. The model for accessing the Calculator can be found in Figure 6-69.

![Figure 6-69: Samsung Task 8 Accessing the Calculator](image)
Problem Analysis for accessing the calculator:

Problem S8a: Again accessing the calculator depends on the sideways menu system. Additionally there is no guarantee that the user will assume the calculator is in the organizer menu. There is also no short cut. Based on this analysis this potential problem is an Visual Appearance and Layout, Simplicity, Directness, Consistency and Familiarity problem.

Formatted Description Part 2: Entering the Equation as in Figure 6-70

Starting at Screen [Calculator]

If the user [decides the equation is finished] the goal is [accomplished].
If the user [decides to wipe an equation], the user must [press] the Clear button which sets the first number, the second number and the answer to [empty], and the first number becomes the [active field]. Control returns to Screen [Calculator]
If the user [decides to get answer], the user must [select equals]. The calculator [sets any empty digits to zero]. The calculator then [calculates the answer]. The answer is set to this [result]

If the user [selects an operator]:
If the first number is [empty], the first number is set to [zero] and the second number is set to active, else
If the first number is [full] and the second number is [empty] the second number is set to [active], else
If the first number is [full] the second number is [full], and the answer is [empty], the calculator [calculates the result], the first number is set to [answer], the second number is set to [empty], and the first number is set to [active]
If the first number is [full], the second number is [full] and the answer is [full], the first number is set to the [answer], the second number is set to [empty], the answer is set to [empty] and the second number is set to [active].

After selecting the operator:
If the plus is [highlighted], it is [not highlighted] and the minus is [highlighted].
Else If the minus is [highlighted], it is [not highlighted] and the times is [highlighted]
Else if the times is [highlighted], the times is [not highlighted], divide is highlighted, If divide is [highlighted], divide is [not highlighted] and plus is [highlighted], screen returns to [enter equation].
There were two obvious problems with entering an equation.

**Problem S8b**: Is the relationship between the text icons and the operator icons. This is a Learnability, Simplicity and Visual Appearance and Layout issue, since these symbols seem unnecessary and only complicate the process.

**Problem S8c**: Unlike the NEC, entering an operator requires cycling through the operators. This is a Directness and Efficiency problem.

The operator selection problem demonstrates the complexity involved with describing some procedures at a low level of detail. Problem S8c could probably have been detected with a simple encapsulated “Move to next operator” function. The option as in NGOMSL is really up to the analyst.
6.4.10 Task 9: Assign voice recording to contact

This task is unique in the sense that no use of voice recognition was used in any of the NEC tasks. In this task the users have to assign a voice recording to one of the records they added in Task 3. This is also the first task to use a recorded voice as a response. The screens involved in this task can be found below in Figure 6-71.

![Figure 6-71: Samsung Task 9 Assign Voice Recording to AL](image)

**Formatted Description Accessing Voice Dial Record**

To access the Voice Dial functions the user must access them through the menu. The Formatted Description for this part is as follows:

**Goal** [Assign Voice Function] from Screen [Idle]
**User** [presses] **Menu Key**. **User** [navigates to Menu Screen Voice Functions]. **User** [decides to use voice recording] **they** must [select Option 3 voice Dial]. This takes them to Screen [Voice Dial Act].

**Problems Accessing Voice Dial Record**

The sideways menu, as in other tasks, is a problem. Navigating to Voice Functions is not straight-forward.

**Problem S9a:** However going to Voice Dial to assign a voice tag is not a logical description, and as such does not match the NGOMSL model. This is therefore a Familiarity and a Directness (poor visual cues) problem.
This however is not the end of navigating, as the user still has to access the screens where the actual recording is initiated. This process is contained in Figure 6-73.

Figure 6-73: Samsung Task 9 Voice Dial

The Formatted Description for actually assigning a voice recording is as follows:

**Goal** [Assign Voice Recording] from **Menu** Screen [Voice Dial]
**User** [decide to record a voice tag] User must [Select Option 1 Record] **Voice Message** replies ["please say name"] an **alert** ["say name"] appears. The **phone** [pauses one second], the **failure count** is [set to 0] and **screen** [Voice dial Recording] appears. **Phone** starts [recording], while the **user** [says name].

If the **recording** [registers no sound] a ["not accepted"] **alert** appears, a **voice message** [says not accepted], and the **user** returns to **screen** [voice dial record] If the **recording** is [heard but rejected] if the **failure count** [is 3], the **phone** has an [error].
If the **failure rate** is [less than 3], the **failure rate** count is [increased by 1] and a **voice message** is played [please say name again]. They are returned to the **alert** [please say name].
If the voice recording is [ok.] A voice recording asks ["Please enter number"]. User should select [phonebook]. Screen [Enter Name] appears. If User [decides name is correct], user must select Find. Phone outputs <contacts matching name> to Screen [find] User [selects contact], user [select view] Screen [View] appears. User [decides to assign Voice Tag], they select [Save]. [Voice Tag set] for selected contact and goal is [accomplished]. If the user wants to [choose another contact]. The user [selects go back] and is returned to [screen idle].

Problem Analysis

Assigning a Voice tag contributed several errors, all at differing levels of seriousness.

The problems are as follows:

Problem S9b: Recording a Voice Tag, is not so obvious, so Predictability and Simplicity are issues

Problem S9c: No feedback if the recording is bad, nor warnings. This involves Forgiveness and Responsiveness

Problem S9d: Contact must be viewed to assign a tag, even though they have already being selected, this is an Efficiency and Predictability issue

Problem S9e: Is a serious Error Recovery issue, if user decides to choose another viewer they are sent back to the start. This is a Predictability and Error Recovery problem.

Most of the problems stem from an over complicated, and long recording process. The lack of support, and the ability to correct mistakes suggests that this task is one of the most difficult in the study.
6.4.11 Task 10: Create appointments in Scheduler

In this task, the subjects had to create an appointment as they did in task 10 of the NEC tasks. Whereas on the NEC the calendar is used to set appointments, it is done using the Scheduler on the Samsung. There is a calendar on the Samsung but appointments cannot be set there. There are several screens the user must navigate to set an appointment on the Samsung. These can be found in Figure 6-74. The users were free to choose, the type of appointment, the only information they were given was the date and time, and the appointment was for lunch.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Daily View</strong></td>
<td><strong>New Appointment</strong></td>
<td><strong>Meeting</strong></td>
<td><strong>Meeting</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Call</strong></td>
<td></td>
<td><strong>14:43</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Meeting</strong></td>
<td></td>
<td><strong>Set Alarm?</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Birthday</strong></td>
<td></td>
<td><strong>Yes</strong></td>
</tr>
<tr>
<td>OK</td>
<td>OK</td>
<td>OK</td>
<td>Yes</td>
</tr>
<tr>
<td>Options</td>
<td>REC</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>5 Set Alarm</th>
<th>6 View Appointments</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Meeting</strong></td>
<td><strong>Meeting</strong></td>
</tr>
<tr>
<td><strong>14:43</strong></td>
<td><strong>14:48</strong></td>
</tr>
<tr>
<td><strong>Set Alarm?</strong></td>
<td><strong>Set Alarm?</strong></td>
</tr>
<tr>
<td><strong>14:43</strong></td>
<td></td>
</tr>
<tr>
<td>OK</td>
<td>OK</td>
</tr>
<tr>
<td>Back</td>
<td>Options</td>
</tr>
</tbody>
</table>

Figure 6-74: Samsung Task 10 Create Appointment
These screens are all accessible from the Organizer menu. Setting the alarm is optional.
Accessing the Organiser follows the same steps as those used in Task 8 to access the calculator. The Formatted Description for this is as follows:

**Formatted Description Part 1: Figure 6-75**

The User’s Goal is [Create Appointment] starting at Screen [Idle]. If User [decides to use menu], user must [Press] Menu Key. User can then [Navigate to Organizer], From Menu Screen [Organizer], if user [decides to use scheduler], they can select Option 1 scheduler. User

**Problem Analysis:**

S10a is the sideways menu problem again; this is a Familiarity, Consistency, Simplicity, Visual Appearance and Layout, and Directness problem.

S10b is a Familiarity and Predictability issue.

The calendar cannot be used to set appointments only the scheduler can.

**Formatted Description Part 2: Figure 6-76**

From Screen [Scheduler] user [selects OK]. If appointments for [day] are [empty], screen [scheduler menu] appears, if [not empty], Appointments for the [day] are outputted to screen [appointments]

User [may select options] or [choose to see next appointment] by [pressing] Nav Up or [choose to see previous appointments] by pressing Nav down and [incrementing] or [decrementing] to the [next] and [previous] appointments for that date and then updating Screen [appointments]. If user is [adding an appointment] they can [select option 1 add], which takes them to screen [scheduler menu]
Figure 6-76: Samsung Task 10: Creating an Appointment

From Screen [Scheduler Menu] User can [choose meeting] and then [press] select. Then from Screen [enter meeting text] after entering meeting text they can [select Ok], taking them to a confirmation that [meeting is saved]. Phone [pauses for two seconds], then from screen [meeting] the user [can choose to set an alarm] if the user [selects no] they return to screen [appointments]. If they [select yes] they can enter Alarm time. Then they can select Ok and are returned to screen appointments.
The diagram above also contains extra functionality for deleting an appointment that is covered in task 12

**Problem Predictions:**

S10c is a Directness and Efficiency problem since the appointments must be accessed one at a time.

S10d is a Directness and Predictability problem since choosing options may not be the next logical step in the sequence to add an appointment. That concludes the creating an appointment task, the major problems both relate to navigation through the system. The entering of data, proved fairly straightforward.

S10e is a potential Forgiveness problem. Since no warning is given before the record is deleted.

S10f is a potential Responsiveness, Predictability and Controllability problem. There is no user action between the confirmation and whichever screen the user is returned to. The message may not be up long enough, or up too long. But the user has no control.

**6.4.12 Task 11: Use Voice Dial to call (in Figure 6-77)**

The formatted description of this task is as follows:

From Screen [Voice Dial], User [wants to make a voice activated call]. User must [select Option2 Voice Act]. Voice Message says [please say name], and Alert [say name] appears. Phone [pauses for two seconds], and failure rate is set to 0 and screen changes to [voice dial record]. User must [say name] and the phone is set to [recording] If the recording has [no sound], the phone [throws an error], alerts user that [recording was not accepted], a voice message says [not accepted], and user is returned to [voice dial record] screen

If the recording is [heard but rejected] and if the failure rate [count = 3], the phone throws an [error] also. Otherwise, the failure rate [count is increased by 1], voice message says [please say name again] and the alert to [say name] is initiated.

If the recording is [ok] a [searching] alert comes up. If phone [finds a matching record], an alert with [name and number] comes up, phone [initiates call], alert says [found number] and Phone [call is started].
Problem Predictions.

S11a: Is a Familiarity issue. Voice act does not convey Voice Dialing, and as such does not match the NGOMSL goal structure.

S11b: Is a Responsiveness and Error Recovery issue. The error messages do not provide any support as to what went wrong. Directness is also a problem.

That concludes the voice dial tasks. The potential problems in both 11 and Task 9, are exacerbated by some strange commands corresponding to the tasks like “Voice Act” and “Voice Command”.

Figure 6-77: Samsung Task 11: Voice Act
6.4.13 Task 12: Delete appointment in Scheduler

The deletion routine task is covered in the task 10 GSE model and so is not modelled here.

Problem Analysis

There are no additional problems in Task 12: Deleting an Appointment.

6.4.14 Task 13: Send saved SMS

In this task the user must send the SMS they created in task 7.

Figure 6-78: Task 13 Send an existing SMS Part 1
Formatted Description for sending an existing SMS part 1: Figure 6-78

Goal [Send Existing SMS] from Screen [Idle] If User [Decides to use key] they [press] the NAV LEFT key and screen changes to [Menu Screen messages], otherwise the user should [press] the MENU key, then [navigate to Screen messages]. The screen changes to [Main Messages Menu].

Formatted Description for sending an existing SMS part 2: Figure 6-79

User [Select Option 3 read message], Screen changes to [Read Messages]. User must [decide to previous message] by [pressing] up button, [next message] by [pressing] down button or [open this message] by [selecting] view. Screen changes to [view message]. User can [move to previous message] by pressing the up button, [the next message] by [pushing] the down button or [sending the message] by [selecting] options. Screen changes to [Options menu]. User [decides to send message], then selects option.

Figure 6-79: Task 13: Send SMS Part 2
Problem Analysis

Problem S13a: It is not obvious that the user should go to Options to send an email. This is a Directness, and Familiarity issue.

Problem s13b: It is not apparent that to access a saved message you must choose read messages. This is a Directness, Familiarity and Predictability issue.

Problem s13c: Cycling through each message one by one is neither Efficient. Nor visually direct.

Problem s13d: Messages are cycled one by one (and not a list) making this neither direct nor Efficient. To open them the user must then select view. This process demonstrates a potential lack of Familiarity and or Predictability.

6.4.15 Task 14: Delete contact from phonebook

![Figure 6-80: Samsung Task 14: Delete Contact]

In this task the user had to delete one of the records created in task 3. This task is only summarised here. Figure 6-79 contains screens involved in this activity. This task was not modelled or analysed in detail.

6.4.16 Task 15: Turn phone off

Task 15, is much like task 1, except that no password is required. Only one potential problem occurs here. Problem S15a corresponds to the time delay before the phone turns off. This is a Responsiveness issue.
6.5 Summary of the Findings

This chapter concludes the GSE modelling for the entire study. The NEC and Samsung devices proved to be challenging environments given the addition of non-traditional tasks. The NGOMSL model, described in Appendix C, demonstrates how user requirements could have been developed as the basis for the GSE model, some of which are converted and displayed in Appendix H. However the NGOMSL task structures proved an excellent reference for shaping the formatted descriptions for the tasks, and allowing the attribute representations to be used to find problems related to issues such as Directness and Familiarity. However to demonstrate that the GSE approach was effective, it is important to discuss the problems found across the two devices.

6.5.1 Samsung

The Samsung GSE models yielded significantly fewer problems than the NEC. This is not surprising, given that number of Samsung tasks was 15, four less than the NEC. However only 27 potential problems were found compared to the NEC’s 51. The most problematic task on the Samsung, according to the GSE analysis, was assigning the voice tags to a contact. There were so many reasons why this particular task could fail, and a lack of useful feedback provided by the phone. Making a phone call, was the second most problematic, however most of these problems related to visual cues
as to what to do. Since the GSE analysis did not address the severity of these problems it will be interesting to see if this task poses significant difficulty to the users in the user tests. Tasks 3: Adding a contact and 4: Using a calculator yielded three problems respectively.

The recurring problem, on the Samsung was found to be the sideways menu, which was an unusual feature. GSE identified this problem due to a lack of visual cues on how to navigate the menu. To identify this problem, the GSE analysis had to model the process at a fairly low level of analysis, much like NGOMSL has to, when working with low level operators. The number of problems found on the Samsung according to each task and the attributes identified can be found in Table 6-3.

<table>
<thead>
<tr>
<th>#</th>
<th>Task</th>
<th>Problems</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Turn Phone On</td>
<td>2</td>
<td>RS EF</td>
</tr>
<tr>
<td>2</td>
<td>Phone Call</td>
<td>4</td>
<td>RS FM DR FG PR EF ER</td>
</tr>
<tr>
<td>3</td>
<td>Add Contact</td>
<td>4</td>
<td>RS FM DR FL DR PR DR PR</td>
</tr>
<tr>
<td>4</td>
<td>Call History</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Access Web</td>
<td>2</td>
<td>VA SM DR CS FM RS ER SM TN</td>
</tr>
<tr>
<td>6</td>
<td>Speed Dial</td>
<td>2</td>
<td>EF CS PR DR EF LN FL</td>
</tr>
<tr>
<td>7</td>
<td>Create SMS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Calculator</td>
<td>3</td>
<td>EF SM FL PR LN SM VA DR EF</td>
</tr>
<tr>
<td>9</td>
<td>Voice Tag</td>
<td>5</td>
<td>FM DR PR SM FG RS EF PR ER PR</td>
</tr>
<tr>
<td>10</td>
<td>Create Appointment</td>
<td>6</td>
<td>FM CS SM VA DR FM PR DR EF DR PR FG RS PR CT</td>
</tr>
<tr>
<td>11</td>
<td>Voice Call</td>
<td>2</td>
<td>FM RS ER DR</td>
</tr>
<tr>
<td>12</td>
<td>Delete Appointment</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Send Saved SMS</td>
<td>4</td>
<td>DR FM DR FM PR EF DR DR EF FM PR</td>
</tr>
<tr>
<td>14</td>
<td>Delete Contact</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Turn Phone off</td>
<td>1</td>
<td>RS</td>
</tr>
</tbody>
</table>

Table 6-3: Samsung potential problems summary

Since the next chapter documents the results of the user studies on the phones with actual users, several questions have been generated based on the problems identified in the GSE analysis.

- Does the delay in turning the phone both On and Off, picked up in tasks 1 and 15, pose a significant problem to the user’s satisfaction?
- When making a call, are the buttons obvious enough? If a mistake is made, will re-entering the number and clearing digits frustrate the user?
- The GSE analysis showed Adding the Contact can’t be done through the phone book. Is entering it in, like making a call, intuitive?
- Is the two-digit speed dial option an efficient means of making a call?
• Is the side-scrolling menu, an intuitive means of navigation? Will users pick it up or will it impede their tasks?
• Is the use of Voice tags, as difficult, error prone, and complicated as the GSE model implies? Secondly, is making a voice activated call of much value, given the GSE analysis showed the process to be long and complicated?
• Is finding the "send" command for SMS messages, going to prove a problem given the GSE analysis demonstrated that it was concealed in the options menu?
• When navigating the calendar will the users have much trouble navigating to the date?

6.5.2 NEC

The problems on the NEC were briefly described in Table 6-2. However, like the Samsung, it is possible to generate some questions on the usability of the phone based on the problems found in the GSE analysis.

• Does the button time delay for turning the phone on and off, complicate this task for the users?
• Does alternate data sources for the phone contacts frustrate the user’s ability to enter and store a new number? Secondly will the strangely inconsistent icon and command structure confuse the users?
• Can the users find calls made, and calls received easily?
• Does the single way of accessing the web, prevent the users from accessing it? Do users understand the status of the connection, and will the users be able to complete their tasks?
• Is the enter button (not found on the NEC) an intuitive means for making a video recording, taking a photo, and attaching a photo to a message?
• Can users find the games, within the unusual Java Menu structure?
• Can users access their saved video recordings, and will the additional preview screens complicate attaching a video image to a message?
• Will users be frustrated by the unusual text field system?
• Will users be frustrated at not being able to enter email messages in the messages section?
• Will, the additional “view” command, when reading a saved text message confuse the user when searching for their text message?
• Will users be able to navigate to the date without difficulty in order to enter an appointment, given the presence of a day view, month view and week view in the GSE models?
• Finally, will the users be able to understand the strange multi-tasking system with its limited feedback, and regular interference in tasks?

The GSE analysis yielded a significant number of potential problems. However will the examiners also find these problems in the user test results? The next chapter
documents the problems found in the user tests on the two phone devices. So will the
NGOMSL based GSE approach proposed in this study be proven to predict an
effective number of usability problems, without access to users, when compared with
user testing? The problems and findings documented in Chapter 7 will help clarify,
which usability problems the GSE analysis is able to find before deployment, and,
which usability problems (found in the user testing) it is unable to. This will reveal
just how effective GSE design and analysis is at finding problems when compared
against post-deployment user testing, considered by many to be the only effective way
to determine the usability of a design.
CHAPTER 7
MAIN STUDY: TEST RESULTS

"Thus the slow and expensive user testing is reserved for those aspects of usability that can only be addressed at this time by empirical trials. If engineering models can be fully developed and put into use, then the designer's creativity and development resources can be more fully devoted to more challenging design problems, such as devising entirely new interface concepts or approaches to the design problem at hand."
(Kieras 1997:734).

7.1 Introduction to the Main Study

The main issue addressed in this chapter, is whether GSE models can be used to predict usability problems prior to user testing. In chapter 2 several usability attributes were proposed that, when violated, contribute to usability problems. Problems attributable to these attributes were then modelled into GSE behaviour trees in Chapter 4. The pilot study in chapter 5, demonstrated that when compared against GSE models of tasks (with a hierarchical NGOMSL like goal structure), a large number of usability problems relating to VCR timer programming found in subsequent user tests, could also be identified. In the previous chapter, a NGOMSL task model of Mobile Phone tasks was built to demonstrate how to establish a hierarchical task model of mobile phone behaviour from scratch. The remainder of the chapter built GSE models of mobile phone tasks based on instructions gleaned from the relevant manuals. Using these GSE models, and the relevant attributes potential usability problems were predicted. It is now time to compare these predictions against actual user tests of the mobile phone tasks, so as to ascertain the success of the GSE approach outlined in this study at finding problems in a far more complicated and interactive environment than that covered in the pilot study.

This second study, conducted using two different mobile phone models, demonstrates just how much effort it can take to undertake rigorous user testing. The various stages of the user test procedure: preparation, acquisition of subjects, conducting the test, using the think aloud data, and analysing the results proved time consuming and costly. However, it is now possible to establish a set of usability problems identified by the three usability experts, and how they compare to the GSE problems specified in the previous chapter.
This chapter presents the results of the main user study, what problems were found, and which usability related attributes contributed to finding them. This chapter is broken into several sections. 7.2 documents the test procedures used to identify the problems, and then compare them against the GSE problems. 7.3 and 7.4 document each of the problems in the user tests for the two mobile phone devices used in the study. These sections also identify the key usability problems for each of these tasks, identified by the three usability analysts. Section 7.5 provides a statistical summary of all the tasks, including an overall comparison of the usability and GSE problems. The last section 3.8 briefly summarises the findings of the main study, and introduces the key points to be discussed in the next and final chapter.

7.2 Test Procedure

Participants:

Participants were recruited from students at various faculties at Griffith University (including the Education, Science and IT faculties), friends, family and colleagues of those already mentioned, as well as staff at an accountancy practice. Approximately 60 participants were supplied with the survey contained in Appendix G. 32 subjects were chosen, the sole criteria being possession of a mobile phone (as gleaned from the surveys). These 32 subjects were paid $20 each to participate in user test sessions taking 1 to 2 hours to complete.

Apparatus and Setting

The setting and room set-up for the user tests, as well as the apparatus used are discussed in detail in sections 4.5.4 and 4.6.2 of Chapter 4: Methodology. These include the two video cameras, phone devices and manuals.

Tasks

The tasks covered are the same mobile phone tasks as those introduced in the previous chapter. The two task sheets for the study may be found at the end of Appendix B. These tasks covered turning the phone on and off, various ways of making a call, creating contacts, appointment creation, using the calculator and several functions specific to each phone. A summary of these tasks may be found in the following table.
<table>
<thead>
<tr>
<th>NEC Tasks</th>
<th>Samsung Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>#</td>
<td>Description</td>
</tr>
<tr>
<td>1</td>
<td>Turn the phone on</td>
</tr>
<tr>
<td>2</td>
<td>Make standard phone call</td>
</tr>
<tr>
<td>3</td>
<td>Add record to contacts</td>
</tr>
<tr>
<td>4</td>
<td>Make a call from number in call history</td>
</tr>
<tr>
<td>5</td>
<td>Access web (Internet) services</td>
</tr>
<tr>
<td>6</td>
<td>Navigate web and download game</td>
</tr>
<tr>
<td>7</td>
<td>Take video sequence using camera</td>
</tr>
<tr>
<td>8</td>
<td>Use calculator to add two numbers</td>
</tr>
<tr>
<td>9</td>
<td>Take a photo of glass</td>
</tr>
<tr>
<td>10</td>
<td>Create an appointment</td>
</tr>
<tr>
<td>11</td>
<td>Compose and send an email</td>
</tr>
<tr>
<td>12</td>
<td>Create an SMS message</td>
</tr>
<tr>
<td>13</td>
<td>Delete the appointment made in task 10</td>
</tr>
<tr>
<td>14</td>
<td>Send photo taken in task 9 to contact added in task 3</td>
</tr>
<tr>
<td>15</td>
<td>Find and start game downloaded in 6</td>
</tr>
<tr>
<td>16</td>
<td>Delete contact made in task 3</td>
</tr>
<tr>
<td>17</td>
<td>Delete Photo taken in task 9</td>
</tr>
<tr>
<td>18</td>
<td>Delete SMS message created in task 12</td>
</tr>
<tr>
<td>19</td>
<td>Turn phone off</td>
</tr>
</tbody>
</table>

**Table 7-1: Summary of tasks in main study user test.**

**Test Procedure**

Subjects performed nineteen Tasks on the NEC and 15 on the Samsung. 16 of the subjects performed the Samsung tasks first, and 16 subjects performed the NEC tasks first. Subjects performed the tasks in individual sessions. Subjects were instructed to express their thoughts aloud, as they conducted the task. They were also informed they were able to ask questions of the researcher relating to clarification of an instruction or any non-task related problems but additional help could not be provided to complete a task. The task sheets (at the end of Appendix B) and instructions were read out at the start of each set of phone tasks. Students were asked if they had any pertinent questions about what they had to do. The sessions then commenced. The test sessions were recorded on the two VCR cameras used in the study. In total, an hour and ten minutes was allowed for using the NEC and forty-five minutes for using the Samsung. When the time allocated for the first phone used was ten minutes from expiring (thirty-five minutes on the Samsung and one hour on the NEC) students were given an additional 10 minutes to complete whatever they were doing, and then begin
the tasks on the other phone. Students were then allowed the remaining time to perform the task on the other phone. Two hours was allocated to each subject such that they had adequate time between the two sets of phone tasks to go to the toilet if necessary. At the end of every task, students were briefly asked to rate each task between 1 (very easy) and 5 (very hard). The examiner noted this down. At the end of the session, subjects were instructed to complete the post study questionnaire, found at the end of their survey forms in Appendix G. The two videotapes were then tagged and stored for later transcription.

**Transcription**

The videotapes, task sheets and transcription instructions were passed on to students hired to transcribe the think aloud protocols into word format. Further information may be found under Protocol Transcription in 4.7.3 of the methodology chapter.

**Problem Analysis**

The procedure for compiling the list of usability problems is virtually identical to that documented in detail in section 5.8. However a summary of this process, as it applies to the main study, is provided here.

The talk aloud protocols (which applied to the six chosen subjects) were passed to three usability examiners. These examiners were usability experts who had all carried out usability testing in the past. They are classified in more detail in section 4.7.2. These examiners (who included the researcher) marked these protocols according to severity (1 to 4) and the attributes they believed applied to a problem. The two external examiners were paid $20 per hour of examination. This was done according to instructions contained in an information pack (see Appendix B). The researcher then collated these marked protocols, into six sets of three protocols, one set for each subject. The various comments were collated into six protocols, containing comments from all three examiners for each of the subjects. Care was taken to mark where comments overlapped on another.

The researcher then entered all the comments into a spreadsheet (Microsoft Excel), and clumps of overlapping comments became potential problem instances. These instances were tabled, including all the attributes and severity information that applied.
to each instance. Next the researcher checked the instances to see what actual problems they corresponded to, by carefully checking them against the protocols focusing on every task, but across all the subjects. Thus a list of potential problems was extracted relating to each task (on each phone). Data pertaining to each problem was carefully organised, such that the instances that made up a problem, as well as the severity, attributes, number of subjects and how many experts noted a problem were also accurately recorded.

The end result was a list of potential usability problems, which could be compared against the GSE predictions for each task. Each user test problem is marked with a task number before the decimal point and a problem number after it. Wherever a problem in the user tests closely matched one described in the GSE models, the user test problem was marked with the relevant GSE problems. Comparison tables were then built containing the user test problem, the corresponding GSE problems and other information as detailed in Table 7-2.

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE Probs</th>
<th>Ex</th>
<th>Sub</th>
<th>Attributes</th>
<th>Pr%</th>
<th>In%</th>
<th>Freq</th>
<th>Sex</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>19.1</td>
<td>Off button duration</td>
<td>N19a</td>
<td>3</td>
<td>1</td>
<td>RS EF DR</td>
<td>100</td>
<td>60</td>
<td>DR EF RS DR</td>
<td>2.67</td>
<td></td>
</tr>
<tr>
<td>19.2</td>
<td>Dislike Phone</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>19.3</td>
<td>Closing Phone</td>
<td>4</td>
<td>1</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>3</td>
</tr>
</tbody>
</table>

Table 7-2: Structure of Problem Tables used in this Study
The next section contains a summary of all the problems found in the user tests, according to each task performed by the subjects in the user test.

### 7.3 Usability problems for the NEC tasks

This section documents the user study results for all 19 tasks, and compares them with those predicted by the GSE models for the same tasks, documented in chapter 6. The first of these tasks involves turning the phone on.

#### 7.3.1 NEC Task 1: Turning the Phone On

There were two significant problems identified in Task 1. The first of these involved finding the actual phone button or finding the number of buttons overwhelming. However, the most obvious problem was the length of time that the user had to press the button before the phone actually started. This problem recurs, in most of the instances above.

**Descriptions of instances**

The six instances in the table above can be described as follows:

1. Subject B found it hard to find the ON Button, and the button was not held long enough.
2. Subject M couldn't get it turned on. The button was not held long enough.
3. Subject B noted that this task was difficult and that no key stood out
4. Subject T did not hold the button long enough
5. Subject T did not hold the button long enough
6. Subject M said there were too many buttons.

50% of the subjects did not hold the button down for the required time. Note that subject T encountered the problem twice, thus only 3 subjects in total were involved.

**Attributes**

Directness appears in all but one of the comments. Two examiners identify it in three and all three examiners do in five. Examiner A reports Familiarity in four instances.

**Comparison to GSE findings**

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE Probs</th>
<th>Sub</th>
<th>Ex</th>
<th>Attributes</th>
<th>Attr</th>
<th>Task</th>
<th>First</th>
<th>Sentence</th>
<th>Res</th>
<th>Ave</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Hold On Button</td>
<td>N/A</td>
<td>3</td>
<td>3</td>
<td>EF L N R S D R</td>
<td>100</td>
<td>55</td>
<td>D R</td>
<td>D R RS</td>
<td>2.71</td>
<td></td>
</tr>
</tbody>
</table>

*Table 7-3: User Test GSE problems for Task 1*
Results were good for this problem that was also predicted in GSE problem N1a. All 3 examiners noted this problem for 50% of the subjects whose talk aloud protocols were used in the study. So this was clearly a problem. Efficiency, Learnability, and Responsiveness were noted and found in both results sets, but Directness proved to be the key attribute in both result sets being the most frequently cited and amongst the attributes rated the highest severity, not surprising given the lack of visual cues as to the time needed to hold the button and activate the phone.

7.3.2 NEC Task 2: Make call to 0407 515 530

In this task the user had to make an ordinary manual phone call, by keying in the phone number and clicking the phone button (P). The average severity for this task was very low (3 and 3.67) and therefore not a critical impediment to the users. In the first problem the user had some difficulty entering the phone numbers to make a call and in the second, the user was not sure how to stop a phone call.

<table>
<thead>
<tr>
<th>No</th>
<th>Description</th>
<th>GSE Prob</th>
<th>Lik. Subj. Attr.</th>
<th>Frq%</th>
<th>Inc%</th>
<th>Start</th>
<th>End</th>
<th>App</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Slow number entry</td>
<td>None</td>
<td>1 1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>3.67</td>
</tr>
<tr>
<td>2.2</td>
<td>Couldn't Stop Call</td>
<td>None</td>
<td>3 1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>3.67</td>
</tr>
</tbody>
</table>

Table 7-4: User Test GSE problems for Task 2

Comparison to GSE findings

Neither of the two problems 2.1 and 2.2 was predicted in the GSE models. Since 2.1 was found by only one examiner and one subject, and has a low severity, it is clearly not a critical problem. The GSE models could not anticipate that one subject would enter the phone number slowly. The second problem 2.2 was also not identified, because cancelling the call was not actually modelled in the GSE task descriptions. But none of the GSE problems N2a, N2b and N2c, which related to problems with the clear buttons, when to enter the number and when to activate the call, were not found in the user test results. Problem N2a was not encountered because subjects only ever-deleted one number a time, and N2b and N2c, were not encountered, possibly because the subjects had access to the support material. GSE problems N2e and N2d were not encountered in the user test results since one is related to the use of a headset and no phone number was re-entered.
The main conclusion is that this task was not a difficult one, and any problems found in the user tests were easily overcome. The problems found in the GSE models were potential problems, but did not manifest themselves in the test. Given the low severity of the instances found by the evaluators, the task was easily understood and executed, and offered little scope for serious usability problems.

### 7.3.3 NEC Task 3: Add to Contacts

In the “Add Contact Task” the subjects had to allocate two names and two phone numbers to the phone book. However since both names were the same, they could either make two records each with the same name but different phone numbers, or make a single record with the one name and two phone numbers. Of the 32 instances 8 distinct problems were determined and listed in Table 7-5.

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE Probs.</th>
<th>Ex</th>
<th>Sub</th>
<th>Attributes</th>
<th>Pr%</th>
<th>Inc%</th>
<th>Freq</th>
<th>Sev.</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Navigation Problems</td>
<td>N3b</td>
<td>3</td>
<td>3</td>
<td>FM DR</td>
<td>100</td>
<td>76.9</td>
<td>DR</td>
<td>DR</td>
<td>2.11</td>
</tr>
<tr>
<td>3.2</td>
<td>Can’t Put Number in</td>
<td>N3f</td>
<td>3</td>
<td>4</td>
<td>RS DR</td>
<td>100</td>
<td>31.6</td>
<td>DR</td>
<td>DR</td>
<td>2.55</td>
</tr>
<tr>
<td>3.3</td>
<td>Identifying Select Key</td>
<td></td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td>0</td>
<td></td>
<td></td>
<td>2.33</td>
</tr>
<tr>
<td>3.4</td>
<td>Wrong Key</td>
<td></td>
<td>3</td>
<td>1</td>
<td></td>
<td></td>
<td>0</td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>3.5</td>
<td>Not familiar</td>
<td></td>
<td>3</td>
<td>1</td>
<td></td>
<td></td>
<td>0</td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>3.6</td>
<td>Use of Enter Key</td>
<td>N3h</td>
<td>3</td>
<td>1</td>
<td>DR</td>
<td>100</td>
<td>75</td>
<td>DR</td>
<td>DR</td>
<td>2.33</td>
</tr>
<tr>
<td>3.7</td>
<td>Menu Problem</td>
<td>N3a N3g</td>
<td>3</td>
<td>1</td>
<td>PR SM, VA DR FM CS PR</td>
<td>42.9</td>
<td>57.1</td>
<td>DR</td>
<td>DR</td>
<td>2</td>
</tr>
<tr>
<td>3.8</td>
<td>Browsing</td>
<td>N3a N3g</td>
<td>3</td>
<td>2</td>
<td>PR SM, VA DR FM CS PR</td>
<td>28.6</td>
<td>100</td>
<td>DR</td>
<td>DR</td>
<td>2</td>
</tr>
</tbody>
</table>

**Table 7-5: User Test GSE problems for Task 3**

The first problem (Navigation Problems), found by all 3 examiners and half the subjects occurred because the subjects had trouble finding the user contacts, which is equivalent to the phonebook on the other phone model. Directness again dominated the attributes in this problem. The second problem (Can’t put the number in) though lower in average severity involved 4 of the subjects (which is the highest percentage of subjects identified for a single problem). In this problem, after entering a name for adding a contact, they were not asked to enter a number but taken instead to the “user details” screen, where entering a number wasn’t readily apparent. Directness and Predictability were the key factors here. Problems 3.5 to 3.6 relate to problems with identifying different keys while 3.7 and 3.8 both involved difficulty using the menu system.
Comparison to GSE findings

5 of the 8 problems manifested themselves in the GSE models 3.3, 3.4, and 3.5, which were not detected related to individual subject’s misconceptions on the function of the keys. Directness dominated all the problems for this task, demonstrating a high severity and frequency across all the problems identified. This demonstrates that visual cues, and the straightforwardness of the operations to be performed are clearly inadequate for first time users. Problem 1, was clearly consistent with that identified in the GSE model, since all the attributes in the corresponding problem N3b, were found in the user test results. 76.9% of the attributes in the user test results consisted of these attributes. Problem 2, the other key problem, in light of the proportion of test subjects who encountered it, was detected in GSE problem N3f. However, the user test results were far more generous with the amount of attributes assigned to this problem, including Predictability Learnability and Simplicity, which were not cited in the GSE model. The GSE models proved successful at picking up problems with the menu system, and the enter key, but not for the other key problems, again demonstrating that the GSE models don’t predict individual subject’s misinterpretation of the individual mobile phone buttons.

7.3.4 NEC Task 4: Activate Call history – find 0407 515 530 and return call using call history

Problem 4.1, was a considered a critical problem, where two users could not find the call history and tried searching ‘user contacts’ first. Directness and Predictability were the key attributes noted by the examiners, but Familiarity and Consistency were also factors suggesting inconsistency with earlier tasks and other phones. Problem 4.3 is a recurrent problem encountered in later tasks as well. In this problem a dialog box appears warning a user that an application (open elsewhere) must be closed for a new application to open. This has to do with the phone’s built-in multi-tasking system, where applications (including standard features like user contacts), must close before another can be opened. The examiners correctly note this as transparency problem given that underlying functionality must be understood by the subject to operate the phone. The only other significant problem (4.2) is one subject being unable to exit the edit screen from the previous task to begin this one.
### Comparison to GSE findings

All three problems (3 out of 3) were found in the GSE models, although problem 4.3, which is a recurring problem, is found in a later task model. Problem 4.1 encountered by half the subjects and all examiners, relates to the unintuitive structure of the menu system. Directness and Familiarity were factors in both result sets, although predictability is only noted in the user test result. Directness is a factor in all 3 task and GSE models, demonstrating its dominance one again. One positive result is that both result sets identified transparency as a root cause of the multi-tasking problem, demonstrating some accuracy in the application of this attribute.

### 7.3.5 NEC Task 5: Accessing ‘3’ Web Services

In this task, users simply had to access the web services menu. Success depended on realising the purpose of the Internet button on the phone. Those who tried to find the web functions via the menu had limited success, while others were thwarted by poor connections.

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE</th>
<th>Ex</th>
<th>Sub</th>
<th>Attributes</th>
<th>Pr%</th>
<th>Inc%</th>
<th>Func</th>
<th>Seve</th>
<th>Ave</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>Are these all web services</td>
<td>N5d</td>
<td>3</td>
<td>3</td>
<td>VA DR RS</td>
<td>66.7</td>
<td>80</td>
<td>RS</td>
<td>TN</td>
<td>2.6</td>
</tr>
<tr>
<td>5.2</td>
<td>No coverage</td>
<td>N5c</td>
<td>2</td>
<td>2</td>
<td>TN RS ER</td>
<td>66.7</td>
<td>66</td>
<td>TN</td>
<td>TN</td>
<td>2.33</td>
</tr>
<tr>
<td>5.3</td>
<td>Identifying the internet button</td>
<td>N5a</td>
<td>1</td>
<td>2</td>
<td>FL DR</td>
<td>50</td>
<td>60</td>
<td>DR</td>
<td>TN</td>
<td>2</td>
</tr>
<tr>
<td>5.4</td>
<td>Can’t get through menu</td>
<td>N5a</td>
<td>3</td>
<td>3</td>
<td>FL DR</td>
<td>100</td>
<td>46.7</td>
<td>DR</td>
<td>TN</td>
<td>3</td>
</tr>
<tr>
<td>5.5</td>
<td>Navigating</td>
<td>N5a</td>
<td>3</td>
<td>2</td>
<td>FL DR</td>
<td>100</td>
<td>50</td>
<td>FL</td>
<td>DR</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 7-7: User Test GSE problems for Task 5

Problem 1 occurred when the web menu appeared, unsure whether they were connected to the Internet or not. Familiarity was the key factor here. Problem 2 occurred because users did not understand what the “No coverage” error message meant. Transparency was the key factor. The third problem occurred when the users were not aware of the Internet button. The fourth problem occurred because web functions could not be accessed via the inbuilt phone menu, with flexibility, predictability and Directness all cited. Problem 5 reflects difficulty subjects had using the menu system.
Comparison to GSE findings

Though none of the problems rated a higher average severity than 2, 80% (4 out of 5) of the problems were predicted in the GSE models. While Directness was a factor in all five problems, Flexibility, Transparency and Responsiveness were also dominant attributes. Problem 2, when connectivity to the Internet is broken and an errors message appears was cited in GSE problem N5c. Transparency was the key attribute and was cited in both models but the two result sets differed on Error Recovery and Directness. Problem 3, relating to the user identifying the internet button was not identified, however GSE problem N5a does relate to this problem given no other way of accessing the web exists. Problem’s 4 and 5, both relate to GSE problem N5a, and the fact the menu does not allow access to the Web. Flexibility and Directness were the key factors in both GSE and user test result sets.

7.3.6 NEC Task 6: Navigate Web services to find Java game: “PlayGolf” and download it.

In this task the subjects had to find the Java game play golf in the web services, and download the game to the phone.

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE Probs</th>
<th>Ex</th>
<th>Sub.</th>
<th>Attributes</th>
<th>Pr%</th>
<th>Inc%</th>
<th>Freq</th>
<th>Severe</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1</td>
<td>Too Complicated</td>
<td>N6a</td>
<td>3</td>
<td>1</td>
<td>SM VA</td>
<td>50</td>
<td>14.3</td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>6.2</td>
<td>No Coverage</td>
<td>N6b</td>
<td>2</td>
<td>2</td>
<td>TN RS ER</td>
<td>33.33</td>
<td>40</td>
<td>TN</td>
<td>TN</td>
<td>2.33</td>
</tr>
<tr>
<td>6.3</td>
<td>Loading Time</td>
<td>N6d</td>
<td>2</td>
<td>2</td>
<td>PR RS</td>
<td>0</td>
<td>0</td>
<td>EF</td>
<td></td>
<td>3.33</td>
</tr>
</tbody>
</table>

Table 7-8: User Test GSE problems for Task 6

Problem 1 occurred because this operation proved to complicated for one of the subjects. In problem 2, two subjects encountered the “no coverage” error preventing completion of the task, until the connection was available. In problem 3, a single flashing icon reports that the download is underway, but provides no indication how much has been downloaded or how long the task will take.

Comparison to GSE findings

Problem 6.1 is equivalent to N6a, where the abundance of visual components, and small size of the display make the task difficult. Simplicity was cited in both result sets, but Directness was the dominant attribute in the user test results, while Visual Appearance and layout was only cited in the GSE model. Problem 6.2 was the same
problem as 5.2 in the previous task and transparency was again the key factor. There was no cohesion here between the attributes cited thus a 0% value for the presence and inclusion of the attributes. GSE problem N6c was not encountered because no download, once initiated, was ever stopped halfway through.

7.3.7 NEC Task 7: Take 12 sec Video Sequence – Cancel Save

In the seventh tasks, subjects were instructed to make a 12 second video sequence (which is the maximum time available) using the phone’s inbuilt camera.

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE Probs</th>
<th>Ex</th>
<th>Sub.</th>
<th>Attributes</th>
<th>Pr%</th>
<th>Inc%</th>
<th>Freq</th>
<th>Severe</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1</td>
<td>Can’t exit Download Game</td>
<td>N6e</td>
<td>3</td>
<td>1</td>
<td>CT FG</td>
<td>100</td>
<td>30.4</td>
<td>CT FG</td>
<td>2.3</td>
<td></td>
</tr>
<tr>
<td>7.2</td>
<td>Button Difficulties</td>
<td>N7a</td>
<td>2</td>
<td>2</td>
<td>DR</td>
<td>100</td>
<td>80</td>
<td>DR</td>
<td>2.5</td>
<td></td>
</tr>
<tr>
<td>7.3</td>
<td>Navigating to Multimedia</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2.83</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.4</td>
<td>Camera Held Incorrectly</td>
<td>Physical</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 7-9: User Test GSE problems for Task 7

In Problem 7.1 a subject had trouble exiting the web functions after downloading the game in the previous tasks. This relates to an earlier GSE problem and is modelled in N6e. In the second problem two subjects did not realise the Enter button started the recording. In 7.3 two subjects did not immediately expect the camera functions to be in the Multimedia menu. The last problem 7.4 occurred because subjects held the camera the wrong way round when taking the video.

Comparison to GSE findings

Only half the user test problems (6.1. and 6.2) were detected in the GSE results. Although the presence of the attributes found in the GSE model, in the user test results was 100%. In problem 1, Controllability and Forgiveness were factors in both result sets, although Directness dominated the user test results. In problem 2, Directness is again the dominant attribute in both result sets. Problem 3 was once instance when a problem was clearly not anticipated. That subjects would not look for the camera functions in the multimedia menu was unexpected. The final problem, which relates to how users orientate the phone, could not be detected in the GSE models. Problem N7b, in the GSE models related to the automatic saving of a video recording and was not detected in the user test problems.
7.3.8 NEC Task 8: Use calculator to add (5 + 2)

In the 8th task, the subjects had to perform a simple addition of numbers in the phone's inbuilt calculator.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>8.1</td>
<td>Slow Navigation</td>
<td>3</td>
<td></td>
<td></td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2.00</td>
</tr>
<tr>
<td>8.2</td>
<td>Identifying Keys N8a, N8c</td>
<td>3</td>
<td>2</td>
<td>VA SM, PR DR</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2.14</td>
</tr>
<tr>
<td>8.3</td>
<td>Picture</td>
<td>3</td>
<td>1</td>
<td>Subjective</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2.67</td>
</tr>
</tbody>
</table>

Table 7-10: User Test GSE problems for Task 8

In Problem 8.1 two subjects had trouble exiting the multimedia functions in the last task. In problem 8.2, 2 subjects had difficulty identifying the keys, which allowed the user to enter the operators (+ and -). In problem 8.3, 1 subject found the on-screen picture of the operators frustrating to interpret.

Comparison to GSE findings

Since exiting the multimedia functions was not modelled in this task, no GSE problem comparable with problem 8.1 was detected. Problem 8.2 was detected in GSE problems N8a, and N8c. Predictability, and Directness were common to both result sets, although Visual Appearance and Layout was not cited in the user test results, despite the apparent confusion in the display. Problem 8.3, a negative response to the key layout on the calculator screen was not predicted, and actually countered by positive feedback expressed by other subjects.

7.3.9 NEC Task 9: Take photo of glass and save

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>9.1</td>
<td>Is it saved</td>
<td>N8a</td>
<td>3</td>
<td>f CT</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4.00</td>
</tr>
</tbody>
</table>

Table 7-11: User Test GSE problems for Task 9

In the 9th task subjects had to take a photo of a glass of water and save it. Only one problem was found here, but of the lowest seriousness possible. Subject K seemed uncertain as to whether the photo was automatically saved after taking it.

Comparison with GSE analysis

Problem 9.1 was identified in the GSE model as a Controllability problem (since the user cannot cancel a save), while the user test results attributed it to Responsiveness and Directness (owing to lack of feedback). So the same problem was identified but
for different reasons. However, the long task sequence identified in GSE problem N9b is not noted in the user test results.

7.3.10 NEC Task 10: Create lunch appointment in scheduler for October 19 2004, 2.30 p.m.

In problem 10, the subjects were asked to add an appointment to a specific date. Problems included subjects being unable to edit an existing appointment, exit the appointment creation screen, get to the scheduler functions, use the soft keys adding the appointment to the phone, identify the time format, navigate to the correct date, and enter and correct appointment information.

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE Probs.</th>
<th>Ex</th>
<th>Sub</th>
<th>Attributes</th>
<th>Pr%</th>
<th>Inc%</th>
<th>Freq</th>
<th>Severe</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.1</td>
<td>Can't Edit Existing Data</td>
<td>Not modelled</td>
<td>2</td>
<td>1</td>
<td></td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>1.5</td>
</tr>
<tr>
<td>10.2</td>
<td>Not being able to exit</td>
<td>N10d</td>
<td>3</td>
<td>1</td>
<td>CT DR PR</td>
<td>100</td>
<td>54.5</td>
<td>DR</td>
<td>DR</td>
<td>2</td>
</tr>
<tr>
<td>10.3</td>
<td>Navigating to Calendar</td>
<td></td>
<td>3</td>
<td>3</td>
<td></td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>2.5</td>
</tr>
<tr>
<td>10.4</td>
<td>Wrong Button</td>
<td></td>
<td>3</td>
<td>2</td>
<td></td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>2.75</td>
</tr>
<tr>
<td>10.5</td>
<td>Adding an Appointment</td>
<td>No Problem</td>
<td>3</td>
<td>2</td>
<td></td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>2.6</td>
</tr>
<tr>
<td>10.6</td>
<td>Slow Day by Day</td>
<td>N10a</td>
<td>3</td>
<td>2</td>
<td>CS DR</td>
<td>50</td>
<td>64.3</td>
<td>DR</td>
<td>DR</td>
<td>2.9</td>
</tr>
<tr>
<td>10.7</td>
<td>Time Formats</td>
<td></td>
<td>3</td>
<td>2</td>
<td></td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>3.2</td>
</tr>
</tbody>
</table>

Table 7-12: User Test GSE problems for Task 10

Comparison with GSE analysis

Only problems 10.6 and 10.2 were clearly identified in the model, although in both these instances, the attributes identified were reasonably consistent with reasonable inclusion values over 50%. In 10.2 the exit option is disabled in the menu, once appointment creation begins, while in 10.3 navigating to the actual date can be a slow and arduous process, if the user does not understand how to switch between day, month and year views. Problem 10.1 was not covered in the GSE models. Problem 5 was not considered a problem in the GSE analysis because it compared well with the NGOMSL sequence. The low number of problems found in this task (2 out of 7) indicates the task requirements for this task were inadequately specified, since none seem beyond the scope of the GSE analysis.
7.3.11 NEC Task 11: Compose and Send Email

In this task the subjects had to compose and send an email from the web services screens.

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE</th>
<th>Ex Sub.</th>
<th>Attributes</th>
<th>Pr%</th>
<th>Inc%</th>
<th>Freq</th>
<th>Severe</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.1</td>
<td>Email and Messaging</td>
<td>N11b</td>
<td>3 4</td>
<td>CS FM DR, PR</td>
<td>100</td>
<td>87.5</td>
<td>DR</td>
<td>CS DR FM PR</td>
<td>2.8</td>
</tr>
<tr>
<td>11.2</td>
<td>Enter Key</td>
<td>N11d</td>
<td>2 1</td>
<td>DR</td>
<td>100</td>
<td>75</td>
<td>DR</td>
<td>DR</td>
<td>2.67</td>
</tr>
<tr>
<td>11.3</td>
<td>Web Navigation</td>
<td>N11a</td>
<td>3 3</td>
<td>DR</td>
<td>100</td>
<td>35.7</td>
<td>DR</td>
<td>DR</td>
<td>2.73</td>
</tr>
<tr>
<td>11.4</td>
<td>Coverage</td>
<td>N5c</td>
<td>2 1</td>
<td>TN RS ER</td>
<td>33.3</td>
<td>50</td>
<td>RS</td>
<td>RS</td>
<td>3</td>
</tr>
<tr>
<td>11.5</td>
<td>Delete Key</td>
<td></td>
<td>3 1</td>
<td></td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>3.67</td>
</tr>
</tbody>
</table>

Table 7.13: User Test GSE problems for Task 11

Problem 11.1 occurred because 4 of the subjects expected an email message could be sent from the phones messaging menu, but this is not possible. Problem 2, relates to activating the on-screen text fields with the Enter button, instead of entering text directly. In 11.3, the email functions are concealed, under the “more” option at one corner of the initial web menu screen. The coverage problem appears again in 11.4, and in 11.5 one subject was confused about the functionality of the C-key.

Comparison with GSE analysis

4 out of the 6 problems were found in the GSE models, although the coverage problem is modelled in earlier tasks. Problems 11.1 and 11.2 demonstrate excellent consistency between the user test and GSE problems, in relation to the attributes chosen. In problem 11.2 the indirect use of the text fields is tightly assigned to Directness in both instances. In problem 11.3 where the email functions are only accessible by accessing the “More” screen was rightly attributed to Directness in both result sets. The coverage problem occurs again, and this time Transparency is not noted in the user test results, demonstrating some inconsistency on the part of the examiners. Problem 5, was not found, but was only attributable to 1 subject, and is considered to be low in seriousness. Another subject expressed some frustration after completing the task but this was not considered a problem.
7.3.12 NEC Task 12: Create SMS message with “Hello” – Save to Memory

In task 12 subjects were expected to create and save a SMS text message. 4 problems were found in this task. These included, the seemingly uneditable text fields on the main SMS screen, knowing whether the message had saved, understanding the function of the delete key, and navigating the save message options.

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE Probs.</th>
<th>Ex</th>
<th>Sub.</th>
<th>Attributes</th>
<th>Pr%</th>
<th>Inc%</th>
<th>Freq</th>
<th>Severe</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.1</td>
<td>Uneditable Text field</td>
<td>N12a</td>
<td>2</td>
<td>1</td>
<td>DR</td>
<td>100</td>
<td>18.2</td>
<td>DR</td>
<td>2.75</td>
<td></td>
</tr>
<tr>
<td>12.2</td>
<td>Save Confirmation</td>
<td>Not modelled</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>DR</td>
<td>2.67</td>
<td></td>
</tr>
<tr>
<td>12.3</td>
<td>Delete Key</td>
<td></td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12.4</td>
<td>Slow Navigation</td>
<td></td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 7-14: User Test GSE problems for Task 12

**Comparison with GSE analysis**

Only 3 of the four user test problems corresponded to GSE models. Problem 12.1 was predicted in GSE problem N12a, and Directness was attributed. Problems 12.2 was not actually modelled, as the requirements used to build the model covered sending and not saving an email. Problem 12.3 and 12.4 was not predicted.

7.3.13 NEC Task 13: Delete appointment in calendar for October 19 2004:

   230 p.m. (Fixed)

In this task subjects were expected to delete the appointment they created in task 10.

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE Probs.</th>
<th>Ex</th>
<th>Sub.</th>
<th>Attributes</th>
<th>Pr%</th>
<th>Inc%</th>
<th>Freq</th>
<th>Severe</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>13.1</td>
<td>Can’t find calendar</td>
<td>N13b</td>
<td>2</td>
<td>1</td>
<td>DR PR</td>
<td>100</td>
<td>60</td>
<td>PR</td>
<td>2.67</td>
<td></td>
</tr>
<tr>
<td>13.2</td>
<td>Thought it was saved</td>
<td></td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>DR</td>
<td>2.67</td>
<td></td>
</tr>
<tr>
<td>13.3</td>
<td>Trouble Cancelling message</td>
<td></td>
<td>3</td>
<td>1</td>
<td>DR PR</td>
<td>100</td>
<td>60</td>
<td>PR</td>
<td>2.67</td>
<td></td>
</tr>
</tbody>
</table>

Table 7-15: User Test GSE problems for Task 13

In problem 13.1 one subject could not find the calendar and scheduler functions again. In 13.2 two subjects thought they had saved the reminder message in task 10, but couldn’t find it. In the last problem 13.3 subjects had problems deleting the appointment.

**Comparison with GSE analysis**

Problem 13.1 was not detected, since this task is only modelled from when they reach the accessories menu. Problem 13.2 relates to task 10, where the save functions were not modelled. Problem 13.3 was predicted, and Directness and Predictiveness were
noted in both result sets with Predictiveness as the dominant attribute in the user test results.

7.3.14 NEC Task 14: Send stored photo to 0412 3848 10:

In this task, the subjects had to send the photo they saved in an earlier task attached to a message. This task yielded a number of problems. In problem 14.1, subjects did not realise when to press the “enter” key to send the photo, since on several keys it was disabled. This problem was significant, given it affected four of the six subjects. Other problems included, navigating the multimedia options, the multi-tasking problem first encountered in 4.2, choosing the image to be sent, trying to send the message via either messages or web services, navigating the multimedia menus and turning the mobile phone’s light off.

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE Probs.</th>
<th>Ex</th>
<th>Sub.</th>
<th>Attributes</th>
<th>Pr%</th>
<th>Inc%</th>
<th>Freq</th>
<th>Severe</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>14.1</td>
<td>Can’t Send Image</td>
<td>N14b, N14c</td>
<td>3</td>
<td>4</td>
<td>CS, CS DR</td>
<td>100</td>
<td>54.8</td>
<td>DR</td>
<td>CS DR</td>
<td>2.19</td>
</tr>
<tr>
<td>14.2</td>
<td>Lots of Navigation</td>
<td>N14a</td>
<td>2</td>
<td>1</td>
<td>DR EF</td>
<td>50</td>
<td>2</td>
<td>DR</td>
<td>DR</td>
<td>2</td>
</tr>
<tr>
<td>14.3</td>
<td>Multi-task</td>
<td>N5c</td>
<td>3</td>
<td>1</td>
<td>TN RS ER</td>
<td>50</td>
<td>66.7</td>
<td>RS</td>
<td>RS</td>
<td>2.33</td>
</tr>
<tr>
<td>14.4</td>
<td>Replace Attached Image</td>
<td>N14f</td>
<td>3</td>
<td>1</td>
<td>ER EF DR</td>
<td>33.3</td>
<td>0</td>
<td>FM</td>
<td></td>
<td>2.67</td>
</tr>
<tr>
<td>14.5</td>
<td>Unsure New Photo</td>
<td>N14f</td>
<td>3</td>
<td>1</td>
<td>ER EF DR</td>
<td>33.3</td>
<td>75</td>
<td>DR</td>
<td>DR</td>
<td>2.67</td>
</tr>
<tr>
<td>14.6</td>
<td>Tries Internet / Messages</td>
<td>N14g</td>
<td>1</td>
<td>2</td>
<td>DR FM</td>
<td>50</td>
<td>2</td>
<td>DR</td>
<td>DR</td>
<td>3</td>
</tr>
<tr>
<td>14.7</td>
<td>Wrong Navigation</td>
<td>N14g</td>
<td>3</td>
<td>1</td>
<td>DR FM</td>
<td>50</td>
<td>60</td>
<td>DR</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>14.8</td>
<td>Light Turning on / off</td>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>4</td>
</tr>
</tbody>
</table>

Table 7-16: User Test GSE problems for Task 14

Comparison with GSE analysis

This task was a difficult one, hence the number of problems, however 7 of the 8 were predicted in some fashion, except for 14.8 when one subject accidentally activated the phone’s light and was unable to turn it off. The first problem 14.2 is detected in N14b, and can be traced to screens that preview the image but can’t attach it to message. Consistency and Directness were correctly identified in both problems. The multi-tasking problem rears its head again (see 4.2) and Responsiveness is the primary cause in both result sets. 14.4, 14.5, 14.6 and 14.7, are attributable to only 2 GSE problems relating to picking multimedia as the menu to send a message with an image, and having to scrap the message to choose another photo to send. Directness was the dominant attribute in most of these problems. No GSE model anticipated the light being turned on.
7.3.15 NEC Task 15: Activate Java Game: Begin game: PlayGolf: Close Java Game

In this task the users had to locate the game they downloaded in task 6 and activate it. When started it pauses and displays a “connecting” message which is odd given the game is already downloaded, and not described as a multiplayer game. This accounted for problem 15.2, while users had trouble finding the location of the game in Problem 15.1, which was stored in the JAVA menu.

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE Probs.</th>
<th>Ex</th>
<th>Sub.</th>
<th>Attributes</th>
<th>Pr%</th>
<th>Inc%</th>
<th>Freq</th>
<th>Severe</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>15.1</td>
<td>Lost</td>
<td>N15a,N15b, N15c</td>
<td>3</td>
<td>3</td>
<td>PR DR, PR RS, PR RS</td>
<td>66.7</td>
<td>56.3</td>
<td>DR</td>
<td>DR</td>
<td>2.82</td>
</tr>
<tr>
<td>15.2</td>
<td>Why does game connect?</td>
<td></td>
<td>2</td>
<td>1</td>
<td>PR</td>
<td>0</td>
<td>0</td>
<td>PR</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

Table 7-17: User Test GSE problems for Task 15

Comparison with GSE analysis

Problem 15.2 was not predicted in the GSE models, since the behaviour of a specific game is not modelled. However, problem 15.1 was detected in GSE problems N15a N15b and N15c and was noted as a Predictability and Directness problem in both result sets, indicating that the JAVA menu was not the logical location for game downloads.

7.3.16 NEC Task 16: Delete all “Al”s from phone Book (contacts)

In this task the subjects had to delete any contacts assigned the name Al in the phonebook, which were added in task 3.

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE Probs.</th>
<th>Ex</th>
<th>Sub.</th>
<th>Attributes</th>
<th>Pr%</th>
<th>Inc%</th>
<th>Freq</th>
<th>Severe</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>16.1</td>
<td>Didn’t Save</td>
<td></td>
<td>3</td>
<td>2</td>
<td>0 RS RS</td>
<td>0</td>
<td>0</td>
<td>RS</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>16.2</td>
<td>Delete all</td>
<td></td>
<td>2</td>
<td>2</td>
<td>0 CT ER</td>
<td>0</td>
<td>0</td>
<td>CT ER</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>16.3</td>
<td>Finding the Contact</td>
<td>N3d</td>
<td>1</td>
<td>1</td>
<td>FG</td>
<td>0</td>
<td>0</td>
<td>FG</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

Table 7-18: User Test GSE problems for Task 16

In problem 16.1, 2 of the subjects could not find the contact they saved in task 3. They may not have saved it properly, or saved it to the USIM card and not the phone’s memory which may be alternated between by accessing a menu option. In problem 16.2 the subjects deleted all the records in the phonebook instead of just the one’s marked ‘Al’ by using the Delete All function. In 16.3 the subject had trouble finding the exact contacts they needed to delete.
Comparison with GSE analysis

Problem 16.1 was dependant on success in an earlier task and not modelled. Problem 2 was not noted as a problem in the GSE models as a warning dialogue was provided. Problem 16.3 related to GSE problem N3d (in an earlier task) that noted the problem with duplicate contact names, but no attributes matched the user test results in this problem.

7.3.17 NEC Task 17: Delete stored Photo

In this task subjects had to delete the photo taken in an earlier task. Three problems were found in the user test, including the multi-task problem found earlier, a subject not being able to confirm if a photo was deleted, and another subject being unable to recall the file name, the image was saved under.

Comparison with GSE analysis

With the exception of the multi-tasking problem found in earlier tasks, there are no GSE problems given this task is not modelled.

7.3.18 NEC Task 18: Delete SMS Message

Only 1 user test problem was found concerning the location of the email, but this activity was not modelled in GSE and no comparison possible.

7.3.19 NEC Task 19: Turn Phone off

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE Problem</th>
<th>Ed</th>
<th>Sub</th>
<th>Attributes</th>
<th>Pr%</th>
<th>inc%</th>
<th>Frmg</th>
<th>Ser</th>
<th>Anal</th>
</tr>
</thead>
<tbody>
<tr>
<td>19.1</td>
<td>Off button duration</td>
<td>N39a</td>
<td>3</td>
<td>i</td>
<td>RS EF DK</td>
<td>100</td>
<td>60</td>
<td>OR EF RS</td>
<td>DR</td>
<td>2.67</td>
</tr>
<tr>
<td>19.2</td>
<td>Closing Phone</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 7-19: User Test GSE problems for Task 19

Three problems were cited in the user tests, the last 2 being ranked low in average seriousness. In 19.1 one subject did not hold the “off” button long enough for it to turn off. An expression of dissatisfaction with the phone in general was noted but was not specific enough or noted by enough examiners to be of comparative value with the GSE models. In problem 19.2, one subject had difficulty closing the phone.
Comparison with GSE analysis

Problem 19.1 corresponded to problem N19a, and both result sets cite Responsiveness, Efficiency and Directness as the root cause. Problems 19.2 and 19.3 relate to a user’s overall opinion and physically closing the phone, and so were not modelled in GSE. Given they applied to only 1 subject and 1 examiner they did not meet the criteria (in Chapter 4) for being significant problems.

That concludes the findings for the NEC tasks. These results demonstrated a NGOMSL based GSE model’s ability to identify major usability problems. Not all problems in the user studies were found, and neither were all the problems in the GSE models found by the user test results. In Chapter 8, these results and their implications are discussed along with those for the pilot study. However only problems concerning one of the two phones used in the main study have been analysed.

7.4 Usability problems for the Samsung tasks

Now that the NEC problems have been analysed, the results for all 15 of the Samsung tasks can be considered.

7.4.1 Samsung Task 1: Turning the phone on.

In Samsung task 1, the subjects had to hold the button long enough for the button to activate, and then enter a password (pin number). This task yielded four problems, relating to finding the on button, working out whether the phone was on, entering the pin number and the length of time it takes to start it.

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE Prob</th>
<th>Attributes</th>
<th>Sub</th>
<th>Ex</th>
<th>Pr</th>
<th>In</th>
<th>Freq</th>
<th>Severe</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>On Button Location</td>
<td></td>
<td></td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>2.67</td>
</tr>
<tr>
<td>1.2</td>
<td>Activation Confirmation</td>
<td>S1a</td>
<td>RS</td>
<td></td>
<td>1</td>
<td>1</td>
<td>100</td>
<td>100</td>
<td>RS</td>
<td>3</td>
</tr>
<tr>
<td>1.3</td>
<td>Enter PIN</td>
<td>S1b</td>
<td>EF</td>
<td></td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>1.4</td>
<td>Long Start-Up Sequence</td>
<td>S1a</td>
<td>RS</td>
<td></td>
<td>1</td>
<td>2</td>
<td>100</td>
<td>20</td>
<td>RS</td>
<td>3.5</td>
</tr>
</tbody>
</table>

Table 7-20: User Test GSE problems for Samsung Task 1

Comparison with GSE analysis.

3 of the four problems were found in both result sets. The location of the ON-Button was a physical characteristic of the phone and not detected in the GSE problems. Problems 1.2 and 1.3 were both responsiveness issues, given trouble with the phone’s feedback and related to GSE problem S1a. Problem 1.3 was also covered, although there was no consensus on which attribute was responsible with the GSE model.
picking Efficiency, and the examiners picked Responsiveness. Given the time delay Efficiency seems more appropriate. There was perfect consensus between the GSE and user test results on problem 1 2. Each user test result in this task was low on seriousness and affected only 1 subject each.

7.4.2 Samsung Task 2: Make call to 0407 515 530

In this task subject had to make a standard phone call. There are no significant problems noted in the user tests. However, the GSE model picked up four potential problems. These included the nature of the clear button (N3c) and the use of the “send” key to make the call (N3d).

7.4.3 Samsung Task 3: Add following names and number to phone book

In task 3, the subjects have to add the same contacts to the phone book as added in the equivalent NEC task. Problems related to not being able to enter the contact from the phone’s phonebook, getting to the phonebook using the phone’s strange sideways menu, accidentally starting a call while entering a number, being stopped by a pre-existing contact, and having to exit the add contact functions before adding another.

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE</th>
<th>Attributes</th>
<th>Sub</th>
<th>Ex</th>
<th>Pr</th>
<th>In</th>
<th>Freq</th>
<th>Severe</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Can’t Put Number In</td>
<td>S3c</td>
<td>DR FL PR</td>
<td>3</td>
<td>3</td>
<td>100</td>
<td>47.6</td>
<td>DR</td>
<td>DR FL PR</td>
<td>2.33</td>
</tr>
<tr>
<td>3.2</td>
<td>Navigating to Phone Book</td>
<td>S5a</td>
<td>VA SM DR CS FM</td>
<td>2</td>
<td>3</td>
<td>40</td>
<td>53</td>
<td>DR</td>
<td>DR FM</td>
<td>2.78</td>
</tr>
<tr>
<td>3.3</td>
<td>Accidentally Start Call</td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>3.4</td>
<td>Pre-Existing Number</td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>3.5</td>
<td>Adding an additional Number</td>
<td>S3d</td>
<td>DR PR</td>
<td>1</td>
<td>1</td>
<td>50</td>
<td>100</td>
<td>DR</td>
<td>DR</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 7.21: User Test GSE problems for Samsung Task 3

Comparison with GSE analysis.

3 out of the 5 user test problems were found in the user test results, while accidentally starting a call, and having to delete a pre-existing number were not predicted in the GSE models. Not being able to put the number in, was attributed to Directness, Flexibility and Predictability in both result sets, given a number could be added only 1 way and not via the obvious location. The second problem, involving the unusual sideways menu recurs many times over the Samsung tasks, and is naturally a Directness problem. Problem 3.5 is so low in serious, and since only 1 examiner and 1 subject notes it, is classified as a minor problem.
7.4.4 Samsung Task 4: Activate Call history

In this task users had to ring an existing number in the phone’s call history. This task yielded only 1 problem encountered by 1 subject and noted by only 1 examiner and with low severity, relating to confusion between calls dialled, calls received and missed calls. No GSE problem is related.

7.4.5 Samsung Task 5: Access Web Browser

In the fifth task, the users had to access the phone’s 3G web services. The first two problems indicate trouble navigating the phone’s sideways menu to the web functions and the last occurred when trying to make the connection.

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE</th>
<th>Attributes</th>
<th>Sub</th>
<th>Ex</th>
<th>Pr%</th>
<th>In%</th>
<th>Freq</th>
<th>Severe</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>Can’t find web option in menu</td>
<td>S5a</td>
<td>VA SM DR CS FM, RS ER SM TN</td>
<td>2</td>
<td>2</td>
<td>33.3</td>
<td>62.5</td>
<td>DR</td>
<td>DR TN</td>
<td>1.4</td>
</tr>
<tr>
<td>5.2</td>
<td>Sideways Menu</td>
<td>S5a</td>
<td>VA SM DR CS FM</td>
<td>1</td>
<td>1</td>
<td>40</td>
<td>100</td>
<td>DR</td>
<td>DR VA</td>
<td>3</td>
</tr>
<tr>
<td>5.3</td>
<td>No Connection</td>
<td>S5b</td>
<td>RS ER SM TN</td>
<td>1</td>
<td>1</td>
<td>25</td>
<td>100</td>
<td>TN</td>
<td>TN</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 7-22: User Test GSE problems for Samsung Task 5

Comparison with GSE analysis.

Problem 5.1 was considered relatively serious by two examiners, and relates to instances S5a (the sideways menu) and S5b relating to how the web functions were set up. Directness and Transparency were dominant attributes here. Problem 5.2 related specifically to the sideways menu noted in GSE problem S5a, and Directness and Visual Appearance and Layout applied in both result sets. Problem 5.3 corresponds to problem S5b, and relates to the ambiguous message given, because no Internet phone number has been set. This was deemed a Transparency issue in both result sets.

7.4.6 Samsung Task 6: Use Speed Dial – Al (0407 515 530)

In this task the users had to access a pre-set phone number assigned to a numeric key. They had to access the contacts in order to determine which key to press. Problems included not being able to use or find the speed dial number and not being sure if the phone had actually started ringing.

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE Prob</th>
<th>Attributes</th>
<th>Sub</th>
<th>Ex</th>
<th>Pr</th>
<th>In</th>
<th>Freq</th>
<th>Severe</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1</td>
<td>Can’t find the number</td>
<td>S6b</td>
<td>DR EF FL LN</td>
<td>3</td>
<td>3</td>
<td>75</td>
<td>62.5</td>
<td>DR</td>
<td>DR LN</td>
<td>1.28</td>
</tr>
<tr>
<td>6.2</td>
<td>Can’t find it in contacts</td>
<td>S6b</td>
<td>DR EF FL LN</td>
<td>2</td>
<td>3</td>
<td>75</td>
<td>62.5</td>
<td>LN</td>
<td>DR</td>
<td>2</td>
</tr>
<tr>
<td>6.3</td>
<td>Not sure if it has been run</td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>2.25</td>
</tr>
</tbody>
</table>

Table 7-23: User Test GSE problems for Samsung Task 6
Comparison with GSE analysis.

2 of the 3 problems, 6.1 and 6.2, are attributable to S6b, since there is no way to check the speed dial numbers without going to the phonebook, and are primarily Directness and Learnability issues. Problem 6.1 (rated at 1.28) is considered to be 1 of the severest problems encountered in this study. GSE problem S6a, concerning the cumbersome entry of two digit speed dial numbers, was not encountered by the subjects, as some did not complete the task and others used one digit speed dial numbers only.

Samsung Task 7: Create SMS and Save.

In this task subjects had to create an SMS text message and save it. Problems included finding the SMS functions, navigating the sideways menu, using the soft and delete keys.

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE</th>
<th>Attributes</th>
<th>Sub</th>
<th>Seq</th>
<th>Fr%</th>
<th>It%</th>
<th>Freq</th>
<th>Severe</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 1</td>
<td>Can't find SMS functions</td>
<td>S5a</td>
<td>VA SM DR CS FM</td>
<td>2</td>
<td>40</td>
<td>50</td>
<td></td>
<td>DR CS</td>
<td>DR CS</td>
<td>1.67</td>
</tr>
<tr>
<td>7 2</td>
<td>Left Right Menu</td>
<td>S5a</td>
<td>VA SM DR CS FM</td>
<td>1</td>
<td>100</td>
<td>100</td>
<td></td>
<td>DR</td>
<td>DR</td>
<td>1.75</td>
</tr>
<tr>
<td>7 3</td>
<td>Understanding Soft keys</td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>7 4</td>
<td>Recognising Delete button</td>
<td></td>
<td></td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>3.33</td>
</tr>
</tbody>
</table>

Table 7-24: User Test GSE problems for Samsung Task 7

Comparison with GSE problems

Only 2 of the 4 problems were found in the GSE models. Problems 7.1 and 7.2, considered serious problems, again apply to the sideways menu encountered in an earlier GSE problem S5a. Problems 7.3 and 7.4 apply to individual subjects coming to grips with the phone controls, and were not detected. Problem 7.4 however, was of low severity.

7.4.7 Samsung Task 8: Use calculator add (5 + 2)

In task 8 subjects had to add two numbers together and produce the answer. There were a lot of problems encountered in this task. These included trouble entering the various operators (6.1, 6.6 and 6.7), navigating the sideways menu (6.2), finding the calculator functions (6.4 and 6.5) and using the soft keys (6.3).
### Table 7-25: User Test GSE problems for Samsung Task 8

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE</th>
<th>Attributes</th>
<th>Sub</th>
<th>Ex</th>
<th>Pr</th>
<th>In</th>
<th>Freq</th>
<th>Severe</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.1</td>
<td>Can't Identify Plus</td>
<td>S8b</td>
<td>S8c LN SM VA, DR</td>
<td>1</td>
<td>3</td>
<td>40</td>
<td>42.9</td>
<td>DR</td>
<td>DR</td>
<td>1.83</td>
</tr>
<tr>
<td>8.2</td>
<td>Unaware of Scrolling Menu</td>
<td>S5a</td>
<td>VA SM DR CS FM</td>
<td>1</td>
<td>1</td>
<td>60</td>
<td>50</td>
<td>DR</td>
<td>DR VA</td>
<td>2</td>
</tr>
<tr>
<td>8.3</td>
<td>Not sure about soft keys</td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>2.5</td>
</tr>
<tr>
<td>8.4</td>
<td>Thinks Calculator elsewhere</td>
<td></td>
<td></td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>2.67</td>
</tr>
<tr>
<td>8.5</td>
<td>Slow Navigation</td>
<td>S5a</td>
<td>VA SM DR CS FM</td>
<td>1</td>
<td>1</td>
<td>20</td>
<td>100</td>
<td>DR</td>
<td>DR</td>
<td>3</td>
</tr>
<tr>
<td>8.6</td>
<td>Trouble entering equation</td>
<td>S8c</td>
<td>DR EF</td>
<td>1</td>
<td>2</td>
<td>50</td>
<td>33.3</td>
<td>EF</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>8.7</td>
<td>Unsure if Calculation completed</td>
<td></td>
<td></td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>3.67</td>
</tr>
</tbody>
</table>

**Comparison with GSE problems**

4 of the 7 user test problems were found in the GSE models. Problem 7.1, encountered by only 1 subject who could work out the plus key, even with the on-screen prompts, corresponds to GSE problem S8c and Directness was the key common factor. 7.2 and 8.5 relate to the sideways scrolling menu problem found earlier in S5a. Those problems not detected applied to 1 subject each except for 8.7 where 2 subjects could not conclude if they had completed the addition. Directness dominated 3 of the 4 problems found in frequency and severity, except for 8.6. 8.6 is an interesting problem, given that Efficiency is the dominant problem relating to having to cycle through all the operators to get to the one the subject needs.

### 7.4.8 Samsung Task 9: Assign Voice Recording to Al (phone book record)

In task 9 users must assign a voice recording to a contact, such that they can make a voice activated call in a later task. Problems occurred when the subject’s voice recording was not accepted (9.1) and navigating the voice options (9.2 and 9.5).

### Table 7-26: User Test GSE problems for Samsung Task 9

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE</th>
<th>Attributes</th>
<th>Sub</th>
<th>Ex</th>
<th>Pr</th>
<th>In</th>
<th>Freq</th>
<th>Severe</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.1</td>
<td>Voice Not accepted</td>
<td>S9c</td>
<td>FG RS</td>
<td>4</td>
<td>3</td>
<td>100</td>
<td>52.4</td>
<td>FG RS</td>
<td>FG DR</td>
<td>1.78</td>
</tr>
<tr>
<td>9.2</td>
<td>Navigation</td>
<td>S9a</td>
<td>DR FM</td>
<td>2</td>
<td>??</td>
<td>50</td>
<td>75</td>
<td>DR</td>
<td>DR</td>
<td>2.5</td>
</tr>
<tr>
<td>9.3</td>
<td>Difficult</td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>9.4</td>
<td>Can't hear voice</td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>9.5</td>
<td>Voice Dial Act Tag confusion</td>
<td>S9a</td>
<td>DR FM</td>
<td>2</td>
<td>2</td>
<td>50</td>
<td>40</td>
<td>DR</td>
<td>DR</td>
<td>3.33</td>
</tr>
</tbody>
</table>

**Comparison with GSE attributes**

3 out of the 5 problems were found in both the GSE and user test results except for 9.3, which related to one subject’s overall difficulty in performing the task, and 9.4
where the subjects couldn’t hear the voice prompts, something that couldn’t be modelled in GSE. The critical problem 7.1 was found in S9e, and given the many times the users voice was not accepted is correctly noted as a forgiveness and responsiveness issue in both result sets. In problem 9.2 and 9.5 the menu options (Voice dial, Voice act etc) were not intuitive for at least 2 of the subjects, with Directness noted in both results sets. However the GSE problem S9a considered familiarity important also.

7.4.9 Samsung Task 10: Create Lunch Appointment in Scheduler for October 19 2004 2.30 p.m.

In task 10, the subject had to create a lunch appointment for a specific date. Key problems found in the user test include unintentionally saving an appointment (9.1), difficulty navigating to the correct date (9.2 and 9.4), and difficulty entering and completing an appointment (9.3).

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE</th>
<th>Attributes</th>
<th>Sub</th>
<th>Ex</th>
<th>Pr</th>
<th>In</th>
<th>Freq</th>
<th>Severe</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.1</td>
<td>Unintentional Save</td>
<td>S10a, S10b</td>
<td>VA SM DR CS</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>FM, FM PR</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.2</td>
<td>Navigation</td>
<td></td>
<td>VA SM DR CS</td>
<td>2</td>
<td>2</td>
<td>71.4</td>
<td>100</td>
<td>DR</td>
<td>DR FM SM VA</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>FM, FM PR</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.3</td>
<td>Difficulty making</td>
<td>S10d</td>
<td>DR PR</td>
<td>1</td>
<td>3</td>
<td>100</td>
<td>75</td>
<td>DR</td>
<td>DR</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>appointment</td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.4</td>
<td>Scrolling to Date</td>
<td>S10c</td>
<td>EF DR</td>
<td>1</td>
<td>1</td>
<td>50</td>
<td>100</td>
<td>DR</td>
<td>DR</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.5</td>
<td>Unexpected Response</td>
<td>S10f</td>
<td>RS CT PR</td>
<td>1</td>
<td>2</td>
<td>33.3</td>
<td>100</td>
<td>PR</td>
<td>PR</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.6</td>
<td>Keys too close</td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.7</td>
<td>Not sure about year</td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.8</td>
<td>Not sure if finished</td>
<td>S10f</td>
<td>RS CT PR</td>
<td>1</td>
<td>3</td>
<td>33.3</td>
<td>20</td>
<td>RS</td>
<td>RS</td>
<td>3</td>
</tr>
</tbody>
</table>

**Table 7-27: User Test GSE problems for Samsung Task 10**

Comparison with GSE attributes

5 of the 8 problems were found in this task were detected in the equivalent GSE models. 10.1, ranked as the most serious was not found, as this was an accidental action performed by 1 subject only. Problem 10.6 and 10.7, which were also not found, related to the physical layout of the keys, and uncertainty as to what year had been chosen (which was not detected in the GSE models). Directness dominated all the problems found, except for 10.8 where the lack of feedback led to Responsiveness being the most frequent and severe when compared to S10f. The sideways menu again caused problems, but so did a very slow date navigation system noted in S10b.
7.4.10 Samsung Task 11: Use Voice Dial to call – [AI]

In this task users were expected to use their pre-recorded voice (in task 9) to activate a phone call. Key problems included, not being able to get voice dial to work at all (considered a severity 1 problem), not being certain it was working, and inadequate support in the manual.

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE Prob</th>
<th>Attributes</th>
<th>Sub</th>
<th>Ex</th>
<th>Pr</th>
<th>In</th>
<th>Freq</th>
<th>Severe</th>
<th>Avig</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.1</td>
<td>Can't Complete</td>
<td>S11a, S11b</td>
<td>FM, RS DR ER</td>
<td>1</td>
<td>1</td>
<td>25</td>
<td>50</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>11.2</td>
<td>Not Certain if Voice Dial Active</td>
<td>S11b</td>
<td>RS ER</td>
<td>1</td>
<td>2</td>
<td>50</td>
<td>50</td>
<td>RS</td>
<td></td>
<td>2.5</td>
</tr>
<tr>
<td>11.3</td>
<td>Manual Not helpful</td>
<td>Man</td>
<td></td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>3</td>
</tr>
</tbody>
</table>

Table 7-28: User Test GSE problems for Samsung Task 11

Comparison to GSE problems

Problem 11.1 was exacerbated by confusion with the menu options (S11a) and a lack of adequate feedback when the process didn’t work (S11b). Despite this Directness was the only attribute shared in both the GSE and user test problems, which is surprising given this does seem to clearly indicate an Error Recovery and Responsiveness problem. Problem 11.2, related to s11b only and this time was noted as a Responsiveness problem in both result sets. 11.3, which was a comment on the quality of the manual documentation could not be predicted in the GSE models.

7.4.11 Samsung Task 12: Delete appointment in Scheduler for October 19 2004 2.30 p.m.

In this task subjects had to delete the appointment they created in task 10. 6 user test problems were reported, but were of low average severity, with 50% (12.4-12.6) rated as minor problems. Problems included subjects hitting the wrong key (12.1), confusing delete by date and delete by day options (12.2), navigation problems (12.3, 12.4, and 12.5) and uncertainty as to whether the record was deleted. However, a low number of subjects were involved in each of the problems noted for this task.

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE</th>
<th>Attributes</th>
<th>Sub</th>
<th>Ex</th>
<th>Pr%</th>
<th>In%</th>
<th>Freq</th>
<th>Severe</th>
<th>Avig</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.1</td>
<td>Wrong Key</td>
<td>S10c</td>
<td>FG</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>2.5</td>
</tr>
<tr>
<td>12.2</td>
<td>Deletion Confusion</td>
<td>S10c</td>
<td>EF DR</td>
<td>1</td>
<td>1</td>
<td>50</td>
<td>50</td>
<td>DR</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>12.3</td>
<td>Navigation</td>
<td>S10c</td>
<td>EF DR</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>DR</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>12.4</td>
<td>Slow Scrolling</td>
<td>S10c</td>
<td>VF SM DR CS FM</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>12.5</td>
<td>Menu</td>
<td>S10f</td>
<td>CT RS PR</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>4</td>
</tr>
</tbody>
</table>

Table 7-29: User Test GSE problems for Samsung Task 12
Comparison to GSE problems

Since the behaviour for this task is modelled in GSE for Task 10, the user test problems are compared with the GSE problems found in task 10. 5 of the 6 problems were found in the GSE models, except for 12.1 where 1 subject kept pressing the wrong buttons. This was not predicted. The others were detected, although the consistency in the choice of attributes between the two was low, with only the navigation problem in 12.2 sharing Directness in common. Problem 12.2 is partially addressed in GSE problem S10e, although a lack of additional information or feedback is also an issue here and that was not noted in the GSE models.

7.4.12 Samsung Task 13 Send saved SMS “hello” to Al

In this task, subjects had to send the text message they saved in task 7 to the contact they created in task 3. Most user test problems related to finding where the message had saved.

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>GSE</th>
<th>Attributes</th>
<th>Sub</th>
<th>Es</th>
<th>Pr%</th>
<th>In%</th>
<th>Freq</th>
<th>Severe</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>13.1</td>
<td>Does not realise saved to read messages</td>
<td>S13b</td>
<td>DR FM PR</td>
<td>1</td>
<td>1</td>
<td>66.7</td>
<td>100</td>
<td>DR PR</td>
<td>DR PR</td>
<td>1</td>
</tr>
<tr>
<td>13.2</td>
<td>Thinks message in presets</td>
<td>S13b</td>
<td>DR FM PR</td>
<td>1</td>
<td>3</td>
<td>100</td>
<td>62.5</td>
<td>DR PR</td>
<td>DR FM PR</td>
<td>2.25</td>
</tr>
<tr>
<td>13.3</td>
<td>Can’t find message</td>
<td>S13d</td>
<td>DR FM PR EF</td>
<td>1</td>
<td>2</td>
<td>100</td>
<td>60</td>
<td>DR PR</td>
<td>DR EF</td>
<td>1.75</td>
</tr>
<tr>
<td>13.4</td>
<td>Can’t find message</td>
<td>S13d</td>
<td>DR FM PR EF</td>
<td>1</td>
<td>3</td>
<td>50</td>
<td>37.5</td>
<td>PR</td>
<td>PR</td>
<td>1.67</td>
</tr>
<tr>
<td>13.5</td>
<td>Thinks it hasn’t saved</td>
<td>S13d</td>
<td>DR FM PR EF</td>
<td>1</td>
<td>3</td>
<td>75</td>
<td>62.5</td>
<td>DR PR</td>
<td>DR FM PR</td>
<td>2.25</td>
</tr>
<tr>
<td>13.6</td>
<td>Trouble navigating</td>
<td>S10a, S13b</td>
<td>VA SM DR CS FM, DR FM PR</td>
<td>1</td>
<td>2</td>
<td>33.3</td>
<td>50</td>
<td>DR</td>
<td>DR</td>
<td>3</td>
</tr>
<tr>
<td>13.7</td>
<td>Looks through menu for messages</td>
<td>S13c</td>
<td>DR EF</td>
<td>1</td>
<td>1</td>
<td>50</td>
<td>100</td>
<td>DR</td>
<td>DR</td>
<td>3</td>
</tr>
<tr>
<td>13.8</td>
<td>Can’t find saved messages</td>
<td>S13d</td>
<td>DR FM PR EF</td>
<td>1</td>
<td>1</td>
<td>50</td>
<td>100</td>
<td>DR FM</td>
<td>DR FM</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 7.30: User Test GSE problems for Samsung Task 13

Comparison to GSE problems

All 8 problems can be detected in the GSE problems to some degree. 13.2, 13.3, 13.4, 13.5 and 13.8 all relate to S13d, where the message is not displayed until the user chooses the view option. Problems 13.1, 13.2 and to some degree 13.6, all relate to saved messages being saved under the “read messages” option and not in other folders such as presets or drafts. The sideways menu was a problem (13.6) even this late into the tasks and can be related to earlier GSE problems such as s10a. Problem 13.7 occurs because the navigation system for choosing between messages is so slow.
Given the poor visual cues and lack of predictable outcomes, Directness and Predictability dominated the result sets for both problems. GSE problem S18c, concerning how messages were sent was not found in the tasks.

7.4.13 Samsung Task 14 Delete Al from phone Book (contents)

In the second last task, subjects had to delete the contacts they added to the phone book in task 3. Only 1 minor problem was noted relating to how one subject read the task instructions. This problem was not detected.

7.4.14 Samsung Task 15: Turn Phone off

The last task simply involved the subjects turning the phone off. The only minor problem was an exclamation of dissatisfaction with the phone in general, and was therefore not predicted.

7.5 Overall Analysis

The mobile phone user tests yielded significantly more user problems than the VCR tests, given the large number of tasks involved. On the NEC 17 tasks were analysed in detail in GSE. Tasks 17 and 18 weren’t modelled since the number of tasks on the NEC significantly exceeded those on the Samsung, and these tasks covered several

<table>
<thead>
<tr>
<th>Task Description</th>
<th>NEC e606</th>
<th>Test</th>
<th>GSE</th>
<th>Task Description</th>
<th>Samsung SGH-T100</th>
<th>Test</th>
<th>GSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Turn Phone On</td>
<td>1</td>
<td>1</td>
<td></td>
<td>1 Turn Phone On</td>
<td>4</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>2 Make Call</td>
<td>2</td>
<td>0</td>
<td></td>
<td>2 Make Call</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>3 Add Contact</td>
<td>8</td>
<td>5</td>
<td></td>
<td>3 Add Contact</td>
<td>5</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4 Call History</td>
<td>3</td>
<td>3</td>
<td></td>
<td>4 Call History</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>5 Web Services</td>
<td>5</td>
<td>4</td>
<td></td>
<td>5 Web Browser</td>
<td>3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>6 Download Game</td>
<td>3</td>
<td>3</td>
<td></td>
<td>6 Use Speed Dial</td>
<td>3</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>7 Video Recording</td>
<td>4</td>
<td>2</td>
<td></td>
<td>7 Create SMS</td>
<td>4</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>8 Calculator</td>
<td>3</td>
<td>1</td>
<td></td>
<td>8 Calculator</td>
<td>7</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>9 Photo</td>
<td>1</td>
<td>1</td>
<td></td>
<td>9 Voice Recording</td>
<td>5</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>10 Appointment</td>
<td>7</td>
<td>2</td>
<td></td>
<td>10 Appointment</td>
<td>8</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>11 Email</td>
<td>5</td>
<td>4</td>
<td></td>
<td>11 Voice Dial</td>
<td>3</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>12 SMS</td>
<td>4</td>
<td>3</td>
<td></td>
<td>12 Delete Appointment</td>
<td>6</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>13 Delete Appointment</td>
<td>3</td>
<td>1</td>
<td></td>
<td>13 Send SMS</td>
<td>8</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>14 Send Photo</td>
<td>8</td>
<td>8</td>
<td></td>
<td>14 Delete Contact</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>15 Start Game</td>
<td>2</td>
<td>1</td>
<td></td>
<td>15 Turn Phone Off</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>16 Delete Contact</td>
<td>3</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19 Turn Phone Off</td>
<td>2</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Totals</td>
<td>64</td>
<td>40</td>
<td></td>
<td>Totals</td>
<td>56</td>
<td>40</td>
<td></td>
</tr>
</tbody>
</table>

Table 7-31: User Test GSE problems for Task 3
functions explored in earlier tasks. The examiners in the NEC study identified 71 potential problems. 7 of these problems were deemed not to be problems, or were general expressions of dissatisfaction by only 1 subject, and were not used for comparison in this study. Of the 64 problems on the NEC, 62.5% or 40 of them were identified in the GSE models. However the GSE results fared slightly better on the Samsung. After removing 8 potential problems, for the same reason as those removed from the NEC model, 56 problems on the Samsung, 71.43% or 40 of the problems were anticipated in the GSE models. The total number of problems in the user tests and those that matched the equivalent GSE models can be found in Table 7-31.

Severity

Severity seemed to be an important factor, when compared against the user test problems identified in the GSE models. To demonstrate this, problem were ranked by their average severity, that is all severity values for a given problem divided by the number reported for that problem. The severity scale used by the examiners in this study (see Table 4-6.) is divided into four key categories. Category 1 was a problem that prevented the completion of a task, category 2 caused delay and frustration, 3 had minor effect on usability and four was subtle and a possible issue for further development. Since total average severity was used, problems were classified according to the nearest whole number. So problems were classified category 1, if the average severity for that problem was within a range of 0.5 but less than 1.5, category 2 if the average severity existed between 1.5 but less than 2.5 and so on.

<table>
<thead>
<tr>
<th>Range</th>
<th>Closest Rating</th>
<th>Found in GSE</th>
<th>Total Problems</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;1.5</td>
<td>Prevent Completion of a Task</td>
<td>10</td>
<td>11</td>
<td>90.1%</td>
</tr>
<tr>
<td>1.5 to 2.49</td>
<td>Significant delay / frustration</td>
<td>24</td>
<td>33</td>
<td>75.8%</td>
</tr>
<tr>
<td>2.5 to 3.49</td>
<td>Minor Effect</td>
<td>40</td>
<td>66</td>
<td>60.6%</td>
</tr>
<tr>
<td>&gt;=3.5</td>
<td>Subtle</td>
<td>6</td>
<td>10</td>
<td>60.0%</td>
</tr>
<tr>
<td><strong>Totals</strong></td>
<td></td>
<td><strong>81</strong></td>
<td><strong>120</strong></td>
<td><strong>67.5%</strong></td>
</tr>
</tbody>
</table>

Table 7-32: User Test Problems Found in GSE Models According to Severity

When classified this way, the GSE models performed significantly better, at identifying Category 1 and 2 problems (at roughly 90 and 75%) than the numerous minor problems (60.6%) and occasional subtle problem (60%). This suggests the technique is better placed to identify problems of a more serious nature. The only category 1 problem not predicted was during the appointment creation on the NEC
(Problem 10.1) where 1 subject could not exit the appointment screen. In this case the behaviour was not modelled suggesting better statement of sub-goals or requirements might have addressed this problem.

**Attributes**

Like the pilot study, the attributes differed significantly from one another in correlation with the user test results. The presence of attributes in the user test results was almost 10% higher on the NEC, and 4% lower on the Samsung when compared against the pilot study average of 56.85%. However when averaged out across the both devices, the presence of attributes is slightly higher at 59.27%. Thus the GSE models performed slightly better in this study, at finding the attributes contained in the user test results. The inclusion rate, which is how many times the attributes cited by the GSE, problems occur in each user test problem, against the total number of times all attributes are cited in the user test results, is up on the pilot study data. The inclusion rate for attributes for the NEC tasks is up about 10%, and the Samsung inclusion rate is up over 20% against the pilot study inclusion rate of 37.13%. The average inclusion rate across the whole main study has risen to 53.45% meaning GSE problem attributes made up an additional 16% of the attributes cited in the user test results. The larger range of task models, and the number of total attributes employed is perhaps the cause for this improvement in the number of GSE attributes cited in the user test problems.

<table>
<thead>
<tr>
<th>Average</th>
<th>NEC</th>
<th>Samsung</th>
<th>Both</th>
<th>Pilot Study</th>
</tr>
</thead>
<tbody>
<tr>
<td>Presence of Attributes</td>
<td>66.18%</td>
<td>52.16%</td>
<td>59.27%</td>
<td>56.85%</td>
</tr>
<tr>
<td>Inclusion of Attributes</td>
<td>47.85%</td>
<td>59.24%</td>
<td>53.45%</td>
<td>37.13%</td>
</tr>
</tbody>
</table>

**Table 3.3: Presence and Inclusion of Attributes**

As for the individual attributes proposed by the GSE models, their presence in the user test results differed significantly. Directness was the most frequently cited attribute, reported 60 times in the relevant GSE problems and was found in 90% of all cases it was reported. No other attribute was reported as many times. Learnability was the most accurately identified. Configurability, as in the pilot study, was not noted. This is not surprising given none of the tasks, involved the subjects using or setting personal preferences with the devices.
The least accurate (scoring under 50%) attributes were Consistency, Controllability, Efficiency, Error Recovery, Simplicity, Responsiveness and Visual Appearance and Layout. The most accurate (scoring above 50%) attributes were Directness, Learnability, Predictability, Flexibility, Familiarity and Transparency. When compared against the pilot study results, the rates of Controllability, Forgiveness, Flexibility, and Simplicity attributes being found in the user test results had all improved. However, the percentage of Efficiency and Error Recovery attributes cited in both result sets was down significantly. Responsiveness and Predictability prediction rates were also down when compared with the pilot study.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Number of Times Reported</th>
<th>Percentage Found in Main Study (%)</th>
<th>Percentage Found in Pilot Study (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configurability</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Consistency</td>
<td>15</td>
<td>33.34</td>
<td>0</td>
</tr>
<tr>
<td>Controllability</td>
<td>6</td>
<td>33.34</td>
<td>28.57</td>
</tr>
<tr>
<td>Directness</td>
<td>60</td>
<td>90</td>
<td>100</td>
</tr>
<tr>
<td>Efficiency</td>
<td>20</td>
<td>20</td>
<td>71.43</td>
</tr>
<tr>
<td>Error Recovery</td>
<td>10</td>
<td>10</td>
<td>28.57</td>
</tr>
<tr>
<td>Forgiveness</td>
<td>4</td>
<td>50</td>
<td>33.33</td>
</tr>
<tr>
<td>Flexibility</td>
<td>6</td>
<td>66.67</td>
<td></td>
</tr>
<tr>
<td>Familiarity</td>
<td>26</td>
<td>53.85</td>
<td>N/A</td>
</tr>
<tr>
<td>Learnability</td>
<td>4</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Predictability</td>
<td>21</td>
<td>57.15</td>
<td>70</td>
</tr>
<tr>
<td>Responsiveness</td>
<td>21</td>
<td>47.62</td>
<td>75</td>
</tr>
<tr>
<td>Simplicity</td>
<td>16</td>
<td>31.25</td>
<td>0</td>
</tr>
<tr>
<td>Transparency</td>
<td>7</td>
<td>71.43</td>
<td></td>
</tr>
<tr>
<td>Visual Appearance and Layout</td>
<td>16</td>
<td>37.5</td>
<td></td>
</tr>
</tbody>
</table>

Table 7-34: Attributes Reported and Found in Main Study
Blank spots indicate attribute is not reported.

The results demonstrate that some attributes (such as Directness) bear out reasonably consistently, whereas others are less likely to be noted in the result of the user tests. However given the rate of problems found in both the GSE and user test models, the attributes proved to be a useful tool in finding them. Further work, better defining the attributes (based on the user test results) may be one way of raising the chance of finding these attributes in the user test results also.
In terms of Severity and Frequency rates for individual attributes, Directness again dominated the results. Directness was noted as the most frequent (49 times) and most severe (50 times) attribute in the 60 times it was cited. The next attribute was Responsiveness, noted 10 times and 9 times for frequency and severity, out of the 21 times it was cited.

Predictability was recorded 8 times for both severity and frequency out of the 21 times it was recorded. Transparency however was the most frequent and severe attribute cited in 57.15% of all the times it was cited. Thus, though the inclusion and presence rates for Responsiveness and Predictability are quite low, the number of times they are cited, along with their severity and frequency indicates that they are the most influential attributes in the main study user tests, across all the tasks.

Problems not found

It is important to review the reasons for why some user test problems were not found. Some of these reasons are as follows:

- **Physical Design Issues:** This included the physical layout of the keys being too close, and the location of the ‘On’ button, or how to physically close the phone.
- **Unforseen Accidents:** This included accidentally starting a call turning on the physical light or saving something accidentally.
- **Familiarity with the Keys:** The user’s perception of the “Soft Keys” and “Delete Keys” on both phones was a recurring problem.
- **Goal Assumptions:** Sometimes it was assumed in the various GSE models that users would logically go to a particular menu for a particular function. This proved to be a mistake, especially with the multimedia and accessories menus.
- **Unforseen considerations:** Such as 24 hour and 12 hour time formats.
• Mistakes from earlier tasks: Some problems compounded, when a file in an earlier task was not saved and available for the next task. In some other instances exiting the functions from a previous task made the next more difficult to begin.

• Expressions of Dissatisfaction: Sometimes the users expressions of dissatisfaction did not relate to one particular task feature.

• Other Design Factors: Factors such as the volume of the voice prompt during the voice recognition tasks were not predicted.

• Unpredicted Behaviour: Such as the subjects decision to delete all the contacts in the phone book instead of just one.

• Incomplete GSE models: In other situations behaviour was not modelled thoroughly enough for problems to be detected.

Some of these problems are oversights or assumptions made by the analyst. Some are limitations that need to be explored in further studies. Other such as 'mistakes from earlier studies' was a direct result of which tasks were chosen and the decision to model these tasks separately.

**Major Design Issues on both phones:**

Though this study was aimed at putting GSE models and attribute prediction through its paces, it is appropriate to summarise the major design issues detected across the two phones. On the NEC these were.

• The multi-tasked applications and web-connectivity problems.
• Trouble turning the Phone on and off due to the time delay.
• Difficulty with adding Contacts.
• "Attaching the Image to an SMS message" and the confusing preview windows.
• The un-editable text fields, and the use of the enter button.

On the Samsung the design issues included:

• The complexity of the voice tag tasks, including the inefficiency of making a voice activated call.
• Difficulty with the side scrolling menu
• Trouble selecting the operators in the calculator.
• Not being able to add a contact from the phone book.
• The inefficiency of the speed dial activities.
• Trouble accessing previously saved SMS messages.

Thus the study identified a range of design issues, including those that were small in scope and task specific to major user interface design and technical issues. This study can only speculate about what might be done to re-design these phones. However, the GSE models have demonstrated their potential as means of predicting significant design issues without user testing.

7.6 Summary.

This main study demonstrated several things. Firstly that GSE models can be used to predict a significant number of usability problems. Secondly, that these problems tend to be among the most severity. Thirdly that several key attributes, particularly Directness are associated with most of the problems. Fourthly that GSE models tend to propose less attributes as the cause of an individual user test problem, than those cited by the user test examiners. Fifthly that at least some of the attributes considered the most severe or frequently associated with a problem, are identified in the overwhelming majority of user test problems. However, there is still significant work to be done, honing the attributes to mirror user test findings more precisely. There are also a significant number of problems not detected in the GSE models, and further exploration of how attributes contribute to problems in GSE is an important basis for further research.
CHAPTER 8
DISCUSSION OF THE RESULTS

"...Even though not all members of software development team may be involved directly in the evaluation of usability, it is useful to have an appreciation of the methods for evaluating usability, particularly practical methods that can be applied with limited skills and resources."
Perlman (1996: 348)

8.1 Introduction

The stated aim of this study was to develop an integrated approach to system design based on cognitive modelling and established usability principles, in order to better predict usability problems. To do this, the study proposed a unique consolidation of techniques. The first of these was GSE, a software engineering notation based on behaviour trees. The second was NGOMSL, a natural language task analysis technique based on Card's (1983) cognitive principles. By using NGOMSL as the statement of requirements it became possible to represent the goal-based behaviour of the user, and the behaviour of the system in a 'total' model of the system. Since the new models are based on NGOMSL principles, this new technique was dubbed NGOMSL-GSE. It is important to stress that NGOMSL was chosen, instead of other GOMS techniques or HTA, for its natural language task orientated format and its compatibility with GSE.

However, this was only the first part of the proposal. In order to prevent usability problems this approach needed to identify them. After examining design principles related to usability, several attributes were identified on the basis that they contributed to the overall usability of a product and when poorly applied could lead to usability problems. Using these attributes, corresponding GSE representations were modelled, as a reference for identifying potential usability problems in a design. The advantage of this approach was that no actual users were needed, and the device did not need to be in a deployed state.

To test the effectiveness of this approach, a choice was made to test GSE models on two older devices (VCRs) employing a notoriously difficult task, and on two ubiquitous devices (mobile phones) with a wide range of new and unusual
capabilities. To prove that usability problems could be found before deployment, the NGOMSL-GSE approach was tested against user tests conducted on the same devices. Then the problems found in the NGOMSL-GSE models were compared against those found in the pilot and main studies, demonstrating that this approach is able to identify a number of usability problems when applied.

This chapter is divided into several sections. The first section 8.2 outlines the outcomes and achievements of the study, against those proposed in the first chapter. The second section 8.3 discusses the results and findings of the studies. This discussion includes which usability problems were found in the NGOMSL-GSE analysis, and how these results compare to the findings of the user tests. The third section 8.4 compares the NGOMSL-GSE technique against criteria specified by Dromey (2002a) and Lim and Arnowitz (2002) for evaluating and choosing task analysis techniques. Section 8.5 discusses future work concerning the approach proposed in the study. The final section 8.6 draws a final conclusion concerning the completion of this thesis.

8.2 Outcomes of this thesis

The objective of this thesis was to build an integrated design technique, capable of predicting usability problems prior to user testing. This was achieved by using a ‘total’ model of system behaviour, which better represents user and system behaviour together, and then applying quality related attributes (represented in GSE) to find these problems. There were three aims, relating to the attributes themselves, the integrated design technique and the conversion of NGOMSL into GSE. This section discusses these aims and whether they were met.

The establishment of a set of quality attributes, modelled in GSE, in order to better identify usability problems before user testing.

This study adopted the view that the eventual usability of a product is dependent on the integration of principles in the design. Unlike guidelines, which are either too vague, or too general in application, this study has defined formal reusable representations of violations of these principles.
This study used behaviour representations for sixteen attributes including Forgivability, Directness and Controllability amongst others. These were modelled in GSE notation (based on NGOMSL goals) to generate useful representations of how violations of these attributes might be found in the design models used in the study. A high proportion of usability problems were found in both studies (88.46% in the pilot study, and 62% and 71.43% in the main study) demonstrating that applying these attributes proved reasonably successful at pinpointing usability problems. However the attributes differed in frequency, severity and consistency when compared between the GSE and user test results.

The results show that while Directness, Efficiency, Familiarity, Predictability, and Responsiveness proved to be useful indicators for problems in both studies, some like Simplicity were less effective and Configurability wasn’t noted at all. Some like Learnability and Directness were found with a high degree of consistency (90% and 100% in the main study) while Simplicity, Consistency and Controllability were not as consistent with the user test results. Thus it can be concluded that the attributes were enormously successful at pinpointing problems. The study also revealed that some attributes related to problems more frequently than others, suggesting that the models and definitions of the lesser-used attributes may benefit from further consideration and study.

The proposal of a new integrated design technique based on cognitive and task modelling principles, and software engineering representations.

The study has demonstrated that the majority of usability problems in both the pilot and mobile-phone studies are identifiable when compared to the attribute representations in Chapter 4. This technique can be applied in two different but important ways. In the first instance, the designer may build a NGOMSL design based on his or her own task requirements, then convert the model to GSE based on the processes document in chapter 2. In the second instance, the designer may apply an NGOMSL model (as in the phone study) to an existing device. In both instances the designer can then compare these models against the formal representations of the usability problems in chapter 4, to eliminate usability problems from their designs. In both studies 3 and 4, the number of usability problems that could be identified using the NGOMSL-GSE technique proved sufficiently high.
The main study demonstrates that the NGOMSL goal structure and its underlying principles yield an appropriate statement of requirements for a GSE model. The advantage of the natural language format of NGOMSL means that the task analysis is neither too difficult to compose nor understand. NGOMSL is equally capable of representing a more complex extensive set of goals and methods (as in the Mobile Phone Model), as it is to represent simple operations such as setting the clock. This study applied it to timer recording a program, making a call, voice activated calls, setting appointments, making calculations, downloading web applications, video recording and other varied tasks. This was accomplished across several devices, including narrowly focused devices (the VCRs) and those more ubiquitous in scope (the mobile phones). Where different devices were analysed on the basis of a common goal, (as in the mobile phone model) the task model proved no less applicable on either of the two.

Despite the success of the study at applying these models to find problems, it is important to ask whether other types of model might have proved just as useful at finding problems. However, other techniques that model cognitive architectures like ACI-R are so specialised that they do not adequately represent system components, system states, data flow, and other issues relevant to design. Another goal-based task modelling technique such as HTA could conceivably be used as the basis for a GSE model. However, HTA relies on annotations for understanding the sequence of events involved while the major constructs in NGOMSL namely, Goals, Objects Methods and Selection rules, are self contained and highly compatible with a behaviour tree structure. An additional benefit is that NGOMSL is clearly grounded on cognitive principles and its other capabilities such as predicting learning time and memory load may also be a rich basis for future work on the GSE approach.

It is the simple fact that all the relevant task behaviour across all the devices could be modelled in GSE that satisfies this second aim of the study. Some problems could not be found, and there were problems such as physical issues (such as the size of the phone keys in the main study) and general expressions of dissatisfaction, which could not be identified. Thus user testing proved capable of finding issues, not able to be modelled in this approach. However the approach did find a majority of the problems
in all studies and the main study results show that this approach found 90% of the most serious. That alone proves the GSE approach proposed by this study is an excellent predictive tool.

The definition of a transformation technique for using a NGOMSL task analysis and its goal based structure as a way of building GSE models that represent both the behaviour of the system and the user.

This study is the first to establish a conversion method between NGOMSL and GSE task models. This is important because NGOMSL, like other GOMS methods is limited in its representation of system components, on which the completion of the user’s goals also depends. It has been demonstrated in Chapter 6 that GSE can formally represent many other important details such as system states, data flow, and even screen contents. Unlike other engineering representations such as UML, GSE’s behaviour trees also adopt a flexible hierarchical structure, which is highly compatible with NGOMSL’s sequence diagrams. This is a practical outcome as other design representations like UML are less suited for this type of model.

The conversion rules also benefit the modelling of user behaviour, since NGOMSL is structured according to its underlying cognitive principles and goal structure. Since the GSE process begins with task analysis (after obtaining the requirements), NGOMSL’s goal orientated task structure is of immense value when the user’s goals and needs are important. The models constructed in Chapter 5 demonstrate that not only can this be achieved with relative ease; it is also possible to identify the relevant system components and their states from the natural language requirements. Given that traditional system design can be impeded by a lack of attention to HCI issues, and task requirements, beginning the design process with an effective set of natural language requirements is of significant value.

By integrating the users’ goals into existing task processes, Chapter 6 demonstrates the cohesion with which NGOMSL may be used with existing system processes. Keeping the user’s goals integrated with more detailed system behaviour, a total system design can be established where designers can both address the technical aspects of their system without losing sight of the users intentions. Ultimately
however, the NGOMSL-GSE approach and its integrated behaviour tree structure, proved an effective design model and representation for locating usability problems.

<table>
<thead>
<tr>
<th>#</th>
<th>Outcome</th>
<th>Yes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The establishment of a set of quality attributes for predicting usability problems</td>
<td>✓</td>
</tr>
<tr>
<td>2</td>
<td>The proposal of a new integrated design technique based on cognitive and task modelling principles, and software engineering representations.</td>
<td>✓</td>
</tr>
<tr>
<td>3</td>
<td>The definition of a transformation technique for using a NGOMSL task analysis and its goal based structure as a way of building GSE models that represent both the behaviour of the system and the user.</td>
<td>✓</td>
</tr>
</tbody>
</table>

**Table 8-1: Outcomes of the Study**

**After the NGOMSL-GSE process**

Once the process document and evaluated in this study is completed. The designer, may correct these task requirements, and may then conduct the entire GSE process, to build a full system model, to determine component dependency and then project the design behaviour trees for each individual component, before moving on to the other stages of the software life cycle. Since the tree structure is the same for both the NGOMSL and system components, there is no reason why they cannot remain alongside each other until the last stage of the GSE process. The designer and system engineer can choose to encapsulate more detailed processes, relevant to their tasks, in the trees

**8.3 Usability problems and findings of the studies**

The results of the two studies demonstrate the effectiveness of NGOMSL-GSE at identifying potential problems. Based on these results it is possible to make the following findings.

**Finding 1: That the NGOMSL GSE method was able to find almost all the problems identified in the user tests.**

In the Pilot Study conducted on the VCR devices, NGOMSL-GSE picked up the majority of the usability problems yielded by the user tests. These included all the problems related to timer recording mode, the clock not being set, and having to cancel the data to correct it. Only two problems were not found. The first of these was
an external problem (B8) where a subject had trouble finding the light on the bar code scanner, and the second (R1/B2) related to the user’s task instructions.

In the main study, again a large majority of the potential problems were identified. These included all the major problems, such as the sideways menu on the Samsung and the multi-tasking problem on the NEC.

Finding 2: NGOMSL-GSE is ill suited to predict external, highly subjective or physical problems.

When comparing the potential problems found in the user tests and in the NGOMSL-GSE models, it is clear that some problems cannot be found in a NGOMSL-GSE analysis. It is possible to classify these problems into five different categories. These are as follows:

- **External factors**: Factors beyond the consideration of the designer.
- **General expressions of dissatisfaction**: When users expressed a general dissatisfaction with a device, this was not always related to a single task.
- Individual and Subjective factors.
- **Physical design problems**: Such as the size of the keys, or the location of the scanning light.
- **Instructional Errors**: Tasks that cannot be completed or at least those that cannot be completed in the manner described in the task sheets.

Finding 3: NGOMSL-GSE was able to find potential usability problems not found in the user tests.

There were a number of problems anticipated in the NGOMSL-GSE model that were not encountered in the user tests. This does not mean that these problems are not issues, which need to be addressed. Instead, many were simply not encountered in the user tests. These included inadequate responses to some types of errors, and problems concerning alternate methods to complete a task.
Finding 4: NGOMSL-GSE can be successfully applied across a range of design contexts.

The pilot and main studies demonstrated that NGOMSL-GSE models could be built for each task on each device. This included the 4 tasks on each of the Panasonic NV-HD620mk2a and NV-SD20 video recorders, the 19 on the NEC e606, and the 16 on the Samsung T100. At no stage could any aspect of the behaviour of the VCR not be modelled.

Finding 5: Depicting sophisticated visual elements and some data transfer in significant detail proved complex.

When modelled in detail, sophisticated screen elements such as animation, and more traditional elements, like highlighted areas on the display, were challenging to include. Additionally some data transfer, and its relationship to other components, such as the timer recording values and the VCR panel in task 4 of the pilot study, became a little complicated. However when modelled at a higher level, this posed no problem. Like NGOMSI, GSE can be explored at multiple levels of detail, and it is to be expected that with increasing levels of detail additional skill may be needed.

Finding 5: NGOMSL-GSE was able to detect problems at differing levels of complexity

Some of the models differed in terms of detail. This was especially so in instances of the VCR tasks where the data types were very carefully defined. It also occurred in the mobile phone models where the relationship between contacts, and the two data storage options (e.g. the USIM and the card). However problems were identified across varying levels of detail, even when some methods were encapsulated.

Finding 6: NGOMSL-GSE was both less time-consuming and less expensive than the time and effort required organising the user tests

Though the task models, did take some time to generate, the natural language descriptions made this relatively simple. Most models could be constructed in half an hour, depending on the task, using Smart Draw version 6. The attribute analysis for each model took an additional ten minutes or so, again depending on the task. The documentation however took significantly longer, but great care was taken to mark and describe these potential problems for the thesis. The time factors were of course
based on a solid knowledge of the task structure of NGOMSL, and the core GSE notation.

However the user tests required considerable time to prepare questionnaires, select subjects, administer the user tests, set up the facilities, transcribe the think aloud protocols, extract potential problems, and draw conclusions. This task was spread across three examiners, six transcribers, and one analyst. It took several weeks to get from the start of testing to tangible design problems. While acknowledging that there are simpler and more automated variations on user testing, the thorough tests conducted in this study were expensive in terms of time, and costs. The alternate option is to hire a usability group to do the work, which reduces the effort but raises the financial costs involved.

In each case, a reasonable amount of expertise was needed. However NGOMSL-GSE required significantly less time, and money to conduct.

Finding 7: NGOMSL-GSE is able to pinpoint problems to lapses in usability related attributes.

The final key finding of the study was that NGOMSL based GSE models were able to identify and describe problems with the design in a more direct manner. For instance, the “cancel to correct” error encountered in the pilot studies was the result of the fields on the VCR not flashing when the field was full. This was detected in the GSE model as a Consistency error, directly targeting the state of the first component and the state [Flashing]. In the user test Consistency was not detected as the cause of this problem at all, and was instead assigned to Forgiveness based on the examiners reports on the user test results. Similar occurrences took place with the Scanner On Off problem, navigating through the appointments in the mobile phone tasks, and making a voice activated call.

Summary of Findings

The list of problems and their associated attributes demonstrated the capabilities of the two testing techniques. The NGOMSL based GSE models, and the representations of usability problems demonstrated that no actual users were needed to find most of the problems in the user tests. A surprising result was the significant difference in
time and expense between the two methods. The NGOMSL and GSE approach is not infallible, and there were several important factors that only the user tests could determine. However, based upon the test findings, the GSE based approach proved to be a competent approach to both modelling the system and identifying significant numbers of usability problems without user testing. It is important also to evaluate this method in relation to external criteria. The next section evaluates the method against established criteria for design and task analysis techniques.

8.4 Evaluating the approach

In order to provide a formal assessment of this method, this section compares the technique against requirements and issues concerning new design methods. This includes requirements for a new method specified by Drome (2002a), and task analysis issues specified by both Lim (2000) and Arnowitz (2000).

8.4.1 Value as a new technique

Drome (2002a) sets six guidelines for new design techniques (Table 8-2). This section discusses each of these in detail.

<table>
<thead>
<tr>
<th>#</th>
<th>Requirements for Justifying a New Technique</th>
<th>Has</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The representation should have well defined formal properties so that its correctness may be explicitly determined.</td>
<td>✓</td>
</tr>
<tr>
<td>2</td>
<td>The representation should make interactive systems easy to represent.</td>
<td>✓</td>
</tr>
<tr>
<td>3</td>
<td>The representation should be modular, so that as much or as little of the device can be represented as desired.</td>
<td>✓</td>
</tr>
<tr>
<td>4</td>
<td>Hierarchical control or structural relations should be easily represented.</td>
<td>✓</td>
</tr>
<tr>
<td>5</td>
<td>The representation should not be committed to any particular hardware or software implementation.</td>
<td>✓</td>
</tr>
<tr>
<td>6</td>
<td>It should be easy to represent features of the system that have psychological implications.</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 8-2: Requirements for a new technique Drome (2002a).

1. Well Defined Formal Properties

The formal structure of NGOMSL, the Goals, Operators, Methods and Selection rules, are preserved in this technique. The NGOMSL construction technique has also been preserved as applied in the main phone study including Kieras's (1997) checks, for Consistency, Effectiveness, and depending on the level of detail, learning time.
(Learnability) also. Using NGOMSL as the basis for a GSE model, the five steps in the GSE method can also be formally applied and tested for correctness. The main components of the GSE method (components, states, events, dataflow etc) may also be tested against Dromey’s own criteria for GSE models.

2. Making interactive systems easy to represent
Since this model incorporates the user as a component alongside many other system components, interaction between the two is formally defined. This is demonstrated in Chapters 5 and 6, where the existing processes for accomplishing the tasks, are converted into behaviour trees that capture the interaction of both the user and the system. The NGOMSL-GSE method transcends traditional GOMS methods where the numbers of system components are limited. The simple nature of the GOMS behaviour tree with its component state basis, has been proved to represent the devices and models in Chapters 4 and 5 with little difficulty even in situations where the interaction was reasonably complicated.

3. Modular Representation
Both NGOMSL and GSE can be explored at any level of detail. Complex operations, that a designer does not wish to elaborate can still be replaced with dummy markers, in NGOMSL and passed on to GSE. Additionally many processes can be encapsulated using the encapsulation notation in GSE, and defined elsewhere if necessary. This is demonstrated in Chapter 6 where many of the methods are encapsulated and many of the operations are limited to a chosen level of detail.

4. Hierarchical control and Structural Relations
NGOMSL-GSE preserves the hierarchical structure of both methods. This is easily demonstrated in Chapter 6, where the hierarchical nature of both the Use Phone method and its behaviour tree representation is preserved. The structural relationships including NGOMSL’s selection rules, methods and decide operators, and GSE’s IF-THEN clauses, iterations, events and conditions are all retained in the integrated model.

5. Not Committed to specific hardware or software
The NGOMSL-GSE technique was applied successfully on both the VCR and mobile phone devices. The number of problems and their correspondence to the attributes
used in the studies demonstrates that the approach found significant usability problems in both instances. Since GSE and NGOMSL can be applied to any task from making tea to assembly lines and satellite technology, and the conversion rules proposed in this study are not based on any specific technology, it is easy to demonstrate the flexibility of the NGOMSL and GSE technique.

6. It should be easy to represent features of the system that have psychological implications

The NGOMSL-GSE approach satisfies this requirement, which is perhaps the most pertinent given the cognitive nature of NGOMSL. Selection rules, user goals and the user’s decision-making processes based on cognitive modelling, and therefore psychological principles, are all retained. Additionally, the cognitive operators that are applied to both working and long-term memory (retain, recall and forget) are relatively easy to apply (as demonstrated in the NGOMSL mobile phone model), with only limited knowledge of the processes. These NGOMSL constructs are all retained in the GSE conversion. User behaviour may also be represented with little difficulty, since the user becomes a component, who acts alongside the other system components involved in the interaction.

8.4.2 Value as a task analysis technique

Since NGOMSL models the user’s tasks and intentions it is a form of task analysis. There are many different task analysis techniques; however these differ in their intent and scope. Now that the studies have been conducted it is possible to evaluate this

<table>
<thead>
<tr>
<th>#</th>
<th>Lim’s Task Analysis Issues from Lim</th>
<th>Has</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Dependent on an existing system.</td>
<td>X</td>
</tr>
<tr>
<td>2</td>
<td>There may be too much analysis.</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>The actual domain (e.g. the workplace) may not be appropriately specified.</td>
<td>?</td>
</tr>
<tr>
<td>4</td>
<td>Outputs of a task are not adequately represented (e.g. what is passed on from one task to the next as part of the workflow).</td>
<td>X</td>
</tr>
<tr>
<td>5</td>
<td>The task may provide insufficient information for selecting and applying different methods to complete a task.</td>
<td>X</td>
</tr>
<tr>
<td>6</td>
<td>The task analysis technique may be poorly developed.</td>
<td>X</td>
</tr>
<tr>
<td>7</td>
<td>The task analysis technique may not be explicitly related to software engineering and system development models.</td>
<td>X</td>
</tr>
</tbody>
</table>

Table 8-3: Issues when choosing a task analysis technique based on Lim (2000:32-33)
technique against criteria specified by both Lim (2000) and Arnowitz (2000). The first set of these issues are summarised in table 8-3.

1. Dependency on an existing system
This was addressed in Requirement 6 of the previous section.

2. Too much analysis
Since designers are free to model behaviour at any level of detail this particular issue is of limited consequence. However, where NGOMSL and GSE models are detailed enough to model primitive operators, and complicated psychological processes this can be an issue. In the pilot study in Chapter 5, the NGOMSL model was reasonably high level and posed little difficulty. In the main study, the devices were explored in much greater detail including cognitive operators. This detail is to be expected when addressing these low level processes. Since it is possible to pre-define some methods, and re-use them across tasks (as in many of the methods in the pilot study) the amount of detail can be minimised. Since the conversion rules apply at any level of detail they do not add additional complexity to the process. As demonstrated in Chapters 5 and 6 once the rules are applied, they need not be addressed again. As for the attributes, they are explored in more detail than they might be if a heuristic was applied, but the aim of this study had been to better quantify them.

3. The domain may not be adequately specified
The context in which the NGOMSL approach is applied is dependent on the person applying it. The environmental issues, which constrain whatever is actually being designed, are beyond the bounds of this study. Whether these are addressed or not is likely to be determined during the requirements elicitation phase, which precedes task analysis. However, there is no reason why any issue can’t be incorporated into a framework that is as flexible as GSE and NGOMSL. GOMS is often criticised for addressing expert and error free behaviour. This study has demonstrated that this need not be the case, as in the “incomplete message” error in the main study and the “cancel to correct” error in the pilot study. In both cases, error-producing behaviour was addressed. However, whether or not the domain is adequately specified by the NGOMSL-GSE technique cannot be answered with any finality in this study. Ironically, it too, will depend on the context in which it is applied.
4. Represent of output
Besides generic methods like the “Set a Value to a Value” method specified in the NGOMSL VCR timer recording model, NGOMSL has only limited capacity for addressing the flow of information. NGOMSL does have three mental operators to represent data held in memory, (Retain, Recall, Forget) though these are limited in scope. GSE however proves well suited for passing information between components. It has four operators (< and > and << and >>) for representing input and output and screen input and screen output. The AS-IS models in Chapter 6 demonstrate how GSE is able to present dataflow. The studies demonstrate that GSE is far better at addressing the flow of information than NGOMSL. However, by converting the NGOMSL methods into GSE, the advantages of the system model are well suited to showing how data passes between user and system such as names and numbers in Chapter 6. The responsiveness attribute identified in both studies, and on all devices, demonstrated that the representation of the flow of information between user and system in GSE can be used effectively to identify possible problems. Thus workflow can be adequately addressed in the NGOMSL-GSE.

5. Sufficient information for decisions and choices
The VCR NGOMSL model in Chapter Five demonstrates how decisions may be organised according to common sense parameters. Both NGOMSL and GSE provide formal structures for decision-making, which may be ordered logically, according to the requirements and user goals. There is no limit to the information that can be used to determine a decision, or on what knowledge this decision is based.

6. Is the technique poorly developed?
The formal nature of the process and its basis on two credible and well-documented techniques counters any claim that the method is poorly developed. The attributes were obtained from several different authors, all of which are established HCI practitioners (e.g. Cooper, Galitz, Shneiderman etc). Though little has changed in the original NGOMSL notation since 1997, variants like the assembly line orientated NGOMSLA, and application of the model across a range of tasks ensures NGOMSL continues to develop. GSE, however, due to its more recent origins, continues to be revised, with both changes to the notation, and its application in new fields. Since
methods are based on sound design principles and extensive source material, it is clear that either technique is far from poorly developed.

7. **Explicit relationship to software engineering and system development models**

The NGOMSL-GSE process is a unification of a cognitive engineering task analysis technique and a behaviour based design and engineering approach. However both techniques are independent of any product or application.

<table>
<thead>
<tr>
<th>#</th>
<th>Task Analysis Issues from Arnowitz</th>
<th>Has</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>Addresses the needs of a team, with members from different disciplines. (e.g. software engineers, and usability experts.)</td>
<td>✔</td>
</tr>
<tr>
<td>9</td>
<td>Can be quickly executed in commercial projects.</td>
<td>?</td>
</tr>
<tr>
<td>10</td>
<td>Provides an easily read overview of the task structure</td>
<td>✔</td>
</tr>
<tr>
<td>12</td>
<td>Provide the designers with assistance and guidance, on how to implement dialogue design.</td>
<td>?</td>
</tr>
<tr>
<td>13</td>
<td>Supply a technique for comparing the task analysis against the concept for the design.</td>
<td>✔</td>
</tr>
<tr>
<td>14</td>
<td>Support the creative aspects of user interface design</td>
<td>✔</td>
</tr>
<tr>
<td>15</td>
<td>Be flexible enough that the task analysis technique can be integrated with other methods.</td>
<td>✔</td>
</tr>
</tbody>
</table>

**Table 8-4: Issues considered when choosing a task analysis technique based on Arnowitz (2000)**

8. **Addressing team and different discipline needs**

The method itself combines cognitive engineering and system engineering, thus already combining two disciplines into the process. The natural language should ensure that any domain specific information could be integrated into the model. Determining how this approach supports all the different disciplines of a development team is beyond the scope of this thesis and an issue for further research. The two studies however, demonstrate that the technique is applicable across device contexts. Thus given flexibility in its application, its support for user and system behaviour, a comprehensive range of additional but consistent notation for those who want to work with a more extensive model, and its independence on any one technology the process supports a range of purposes and domains. Since it can also tackle goal orientated models and detailed system requirements, it is clearly suitable model for a wide range of personnel and their respective purposes.
9. Can be quickly executed in commercial projects
With the NGOMSL-GSE procedure in its infancy and a development tool for GSE under construction, there is strong reason to suggest the method will only get faster. In terms of the first two tests, the time allocated to the analysis was significantly less than that spent preparing the user test, transcribing the think aloud, marking the potential problems and then documenting the results. This suggests that the method should take less time than intensive user testing.

10. Provides an easily read overview of the task structure
The natural language component of the method, demonstrated in both studies, shows that the models are relatively easy to read. In GSE some knowledge will be needed of the basic syntax for components, states, conditions, events, loops, IF-THEN-ELSE sequences, input/output and encapsulation, to fully understand the GSE models. However, the notion is by no means extensive and adheres to the simple behaviour tree structure. Many of the models, especially at a high level of detail need only the basics, but since the idea is to provide a system design, the other notation is probably necessary. The NGOMSL-GSE models in Chapters 5 and 6 are reasonably easy to follow (Chapter 5 especially so). Since everything is contained in the one notation, and this notation is based on natural language requirements, it can be concluded that the model is reasonably easy to read.

11. Dialogue design – guideline and assistance
The GSE process, in its later stages demonstrates how the grouping of components and information together, alongside relevant input and output events, the designer can abstract a great deal about the interface, and screen design. Since this study concentrates on establishing predictive task models against existing devices, it did not explore the entire design process from task elicitation through to prototyping and deployment (since the idea was to prevent problems), therefore this question cannot be adequately answered.

12. Means of comparing task analysis against design concept
The basis for these models was an analysis of the users’ goals. These goals should be based on the design concept, given that some task must be performed. However, it is possible to have alternate designs, in much the same way that the two VCRs and two
Mobile Phones were compared to each other according to tasks. Alternate designs can be modelled in exactly the same way. Attributes can be applied, just as they were in Chapters 4 and 5.

13. Support creativity

This is a difficult one to answer, other than to say that both techniques may be applied to any situation at any level of detail. There are some formal rules for conversion, and building these models but the user is free to apply them to whatever design they wish to examine.

14. Flexible enough to integrate with other methods

There is no quick answer to this question. GSE and NGOMSL fit into one part of the design process. They do not elicit the requirements themselves. As such developers are likely to use methods alongside this process to obtain this kind of information. There are also engineering issues, in later stages of development that other methods may need to be adopted. Also there is no reason to suggest that developers would not choose to use user testing as well as other usability techniques to comprehensively address as many usability issues as possible. There would seem to be no reason to conduct a HTA analysis alongside a NGOMSL analysis, given NGOMSL does everything HTA does and more. This study can only acknowledge that NGOMSL and GSE have proved to be highly compatible, that GSE behaviour trees can model things well beyond software design, but that NGOMSL and GSE do not address every aspect of development process. So the only certain answer is that it will depend on the method.

8.5 Future Work

This thesis has proposed the integration of NGOMSL and GSE modelling, along side quality producing attributes as a means of predicting usability problems. The studies have demonstrated that at the very least, this is an acceptable basis on which to model and integrate the goal-based behaviour of the user and more technical behaviour of the system. Since this is the first time that NGOMSL and GSE processes have ever been integrated, or the usability attributes defined and modelled in GSE, there is a
great deal of scope for further development. Though this study did apply the techniques in a ubiquitous phone environment, interface design continues to evolve to new frontiers, and other usability techniques evolve in response to new technologies like eye tracking. Like any other HCI method NGOMSL-GSE technique proposed in this study, must also address these challenges, along side a number of other issues, which were not addressed in the study.

**More detailed analysis of low level and parallel cognitive processes**
NGOMSL-GSE is applicable at various levels of detail. This main and pilot study focused on NGOMSL goals, control structures, selection of methods, and high-level operators. This focus was suitable to predict the presence of major usability problems, and make the technique easier to apply and describe. However, to demonstrate the value of this technique in addressing more complex cognitive processes (in relation to user behaviour) future work should be conducted on both parallel cognitive processes parallel processes (such as those covered in CPM-GOMS), and low-level operators and metrics (as in the KLM Model).

**The implication of contextual issues**
While NGOMSL-GSE is applicable in almost any given context, its relationship to specific task environments and associated scenarios is beyond the scope of this study. Addressing how contextual issues may be better represented in NGOMSL-GSE is likely to enhance its application.

**Testing of this technique against other methods and environments**
Though this model was tested against a comparative user study there are other methods worthy of examining, including the application of heuristic analysis, guidelines, and other user-testing formats. New technologies such as eye tracking, and variations on other modelling techniques such as CPM-GOMS and ACT-R, should also be compared. Additionally, the expansion of the user interface to new environments, such as other ubiquitous devices like PDAs, and even wearable computing need to be considered given the continual advancement in technology.
Automation

Many existing processes have already moved beyond manual analysis and been adapted into tools (e.g. AVID). The advantages of such a move include improved speed, better ways for building design models, hiding unnecessary detail, and the automatic generation of functional prototypes. GSE is still its infancy, its notation is under frequent revision but it shows a lot of promise as a unified design model.

Given that tools can only complicate the design process if the basis on which they are built is not sound, successful automation of the NGOMSL-GSE model is probably a little way off. However the advantages of speeding up the process, and making it easier for less experienced designers, will undoubtedly determine the future popularity of the process, given that time, experience and expense prevent the adoption of other tools.

Thus, the future of this technique lies in ability to keep pace with the rapid evolution of user interface technology and the changing needs and expectations of designers. However, this thesis has demonstrated the effectiveness of this technique at finding usability problems its soundness as a usability technique. Given this result, there is hope the model while retaining its ease of use, will evolve to meet these challenges.

8.6 Conclusion

Because HCI is so closely tied to cognitive principles, it is likely that it will be some time before many performance issues can be modelled to the satisfaction of every HCI professional. More important at present are the practical implications of contemporary HCI techniques. Passing these techniques on, in an effective form, to designers and developers who may or may not have a grasp of HCI principles and usability design, requires techniques that are themselves “easy to use.” Regardless of whether this is accomplished through automated tools, the underlying technique must be theoretically sound (Van Setten: 1997).

The method proposed in this paper, is still in its infancy, but the results are promising. This GSE (Dromey: 2002a) and NGOMSL (Kiers: 1997) based design method offers a unified process, which if adhered to is capable of preventing many of the problems
identified in the user tests, before deployment. Since it is capable of integrating
cognitive and engineering models into a single format, it offers a promising means to
satisfy the NITRD's (2003) search for a total system model. In any case, this thesis
has proved the technique capable of identifying significant usability problems prior to
deployment, which given the effort and expense associated with user testing, is a
valuable result in its own right.
BIBLIOGRAPHY


Page 393


*Interacting with Computers*, 14, pp 397 – 405


*Interacting with Computers*, 11, pp 525-543


Byrne, M.D. (2001). ACT-R/PM and menu selection: applying a cognitive architecture to HCI. International Journal of Human-Computer Studies, 55, 41-84


CUE : Comparative Usability Evaluation Final Reports,
http://www.dialogdesign.dk/cue.html

CUE-1 (1998) Comparative usability test of a Windows program by 4 teams,
http://www.dialogdesign.dk/texter/cue1/cue1paper.doc

CUE-2 (1998) Comparative usability test of www.hotmail.com by 9 teams,
CUE-3 (2001) Comparative usability test of Avis.com by 12 teams [Scandinavia],
http://www.dialogdesign.dk/texter/cue3/all_cue3_reports.pdfCUE-3 (2001),
Reviewed by Molich, R. [qv.] www.DialogDesign.dk

http://www.dialogdesign.dk/texter/cue4/all_cue4_reports.pdf

CUE-4 Scenario (2004). Usability Evaluation for Comparative Review Workshop

Mahwah, New Jersey: Lawrence Erlbaum Associates: pp 413-427

Danielson, D.R. (2002). Web navigation and the behavioural effects of constantly

Effects of Expertise and Familiarity on the Classification of Object-Oriented Code.
Human-Computer Interaction, 10 [2&3] pp 227-248


Raton CRC Inc.

Neuropsychological Approaches to Mental Imagery, Proceedings of the European
Workshop on Imagery and Cognition, NATO Advanced Research Workshop, Orsay
Nijhoff


http://members.tripod.com/bazman/index.html


http://hfac.gmu.edu/~gray/courses/psyc645/downloads/wk10/NGOMSL/wk10/NGOMSL.ppt


Also at: www.webcollab.com/allee/papers/interact95-goms.pdf


Mayhew, D. J. (1999). The Usability Engineering Lifecycle, San Francisco California: Morgan Kaufmann Publishers


Page 425


http://www.ccic.gov/ac/report/


http://www.eouroundtable.com/


SEDRIS (2003). Environmental Data Representation and Interchange

[http://www.sedris.org/glossary.htm](http://www.sedris.org/glossary.htm)


Page 436


XML [Extensible Markings Language] Specifications, W3C Architecture Domain, [http://www.w3.org/xml](http://www.w3.org/xml)


Manuals


GLOSSARY

Attribute: “an attribute is a given fact, which has been attributed, ascribed, imputed, assigned or otherwise given.” (van de Vruvel, 1996).

Conceptual model: “a high-level description of how a system is organized and operates”. It is an “idealized view of the how the system works” which is “the model designers hope users will internalize”, “the ontological structure of the system” (“the objects, their relationships, and control structures”) and “the mechanism by which users accomplish the tasks the system is intended to support” (Johnson and Henderson 2002: 26 – 27)

End-User Computing: The capability of computer users (and not just information technology professionals and commercial developers) to develop and maintain their own applications and systems as well as the accessing and use of resources to do so (See McGill 1998, Ameroso 1998, Barker and Wright 1997, and Lamb and Davidson 2005).

Heuristic “Recognised usability principles” (Nielsen 2004: 45-9) or rules of thumb “derived from more extensive collections of interface guidelines” (Cockton, Lavery and Woolrych 2003: 1122).

Information Appliance
“An appliance specialising in information: knowledge, facts, graphics, images, video or sound. And information appliance is designed to perform a specific activity, such as music, photography or writing. A distinguishing feature of information appliances is the ability to share information among themselves.” (Norman 1998: 53).

Mental model: The model created by the user in order to understand a system. It may be incomplete and will evolve as the user learns more about a system. It allows them to understand how the system works and determines how they interact with it.

Model: “a physical, mathematical, or otherwise logical representation of a system, entity, phenomenon, or process”... “using some type of formalising consisting of variables and the constraints between these variables.” “A model is also a description of the generic structure and meaning of a system,” an abstraction capturing the essential aspects of a system (depending on the intent) and ignoring some of the details. (from SEDRIS 2003, Vincente 1999, Rumbaugh Jacobson and Booch, 2005: 21).

Model Based Evaluation: “using a model of how a human would use a proposed system to obtain predicted usability measures by calculation or simulation." Kieras (2002: 1140).

Principle: “a rule or code of conduct”

Product Quality – Quality specific to the product in question based on one or more different quality dimensions including, attributes, user needs, consumer preferences, and the value of the product in question (see Garvin 1984).
Property: Either the attributes of an entity type or “thing” (e.g. “the weight” of an object) or a relationship between entities such as “cars owned”. (Parsons and Cole, 2004).

Quality: A set of desirable properties, which expresses the needs and requirements of “different interest groups” or stakeholders. Quality represents conformance to these requirements. (Dromey 1998: 3 and Juran cited in Firquin 1992:3)

Quality Attributes (Qualities): “desirable properties” (Dromey, 1998: 3)

Software Engineering: An engineering focused “approach to software development” or a “systems view of software” that concerns the development process, the requirements, testing, the general design and architectural issues of “not only the software, but the systems that contain the software”. (Mayhew, 1999: 3, Lewin 1988: 1)

Software Product Quality – Product Quality as it relates to any desirable quality attribute associated with a software product (see Dromey 1998).

Standard: “a specification recognized within an organization” (Rada 1996: 19)

System Behaviour: “the set of responses (including outputs and metrics) the system exhibits through the execution and interaction of sets of its functions in response to one or more sets of inputs.” (Dromey 1998:4)

System image: All the components of a system with which the user comes in to contact, essentially all the parts of the user interface. However, it also describes how the user’s computer (the computer on which they are interacting) looks, and the content of any information provided to the user (see Norman 1986).

Trade-off: The sacrifice or weakening of a quality producing principle or attribute because of either, another conflicting quality producing principle or attribute, or project constraints.

Usability: how successfully the system facilitates the user in effectively, efficiently and satisfactorily meeting their goals, through interaction with the system.

Usability Design: Designing a system to usability principles, guidelines, and or standards.

Usability Objectives: Any objective, design rule, principle or attribute that, when implemented, supports the usability of a product.

Usability Reengineering: the process/discipline of using techniques aimed at improving the usability of an existing system.

Usability Evaluation: the use of any method that provides a measurement of usability or seeks to identify usability problems, at any stage of the development process, including usability testing, inspection and model based evaluation.
**Glossary**

**Usability Inspection**: the use of heuristics, guidelines or other criteria to find usability Problems (Ivory and Hearst 2001, Nielsen 1993) without observing actual users.

**Usability Reengineering**: the process/discipline of using techniques aimed at improving the usability of an existing system.

**Usability Testing or User Testing**: is a test involving the actual observation of users, after which the researcher analyses the observations looking for usability Problems.

**User interface (UI)**: “A computer-mediated means to facilitate communication between human beings or between a human being and an artifact. The user interface embodies both physical and communicative aspects of input and output, or interactive activity. The user interface includes both physical objects and computer systems (hardware and software, which includes applications, operating systems, and networks.” (Marcus 2002: 19)

**User Interface Quality** – Quality as it relates to the user interface, based on either the successful facilitation of human-system interaction and or the quality attributes deemed to contribute to this facilitation.
APPENDIX A GSE Behaviour Tree Notation
Behavior Tree Notation - Component-State Forms (Abridged) V.04.1

1. Tag C[s] Component C realizes state s then passes control to its output. Component state boxes are "green" if in green highlight.

2. Tag C[s] System Component C realizes the system state s then passes control to its output.

3. Tag C[n: exp] Component C realizes the state s if its attribute n is assigned the value of the expression, product or state "exp". It then passes control to component connected to its output.

4. Tag C[s] If component C is in state s it passes control to its output. This notation is used to model decision conditions. It does not affect flow of control through the component state is blocked.

5. Tag C[nOR:b] If component C is in state "nOR:b" it passes control to its output. This notation is used to model decision conditions. The set of conditions emanating from a component are orthogonal.

6. Tag C[b? s?] When or if component C has realized state "b" it passes control by the output. This notation is used to model decision. Only one of a set of alternative events can ever be true at one time.

7. Tag C[s] Component C satisfies the state s if attribute value "s" to the component connected to its output.

8. Tag C[s] Component C satisfies the state s if attribute value "s" to the component connected to its output.

9. Tag C[s] State s in this tag field indicates that the behavior corresponding to C in state s is not explicitly related to the original requirement but is implied by the requirement context "yellow" is used for "implication"

10. Tag C[s] The "s" in the tag field indicates that the behavior corresponding to C in state s is relating from the original requirement and is needed for completeness. The color "red" is used for "relevancy".

11. Tag C[s] The "s" in the tag field indicates that the behavior corresponding to C in state s is has been added as the post-development of maintenance frame. The color "blue" is used for F1 states.

12. Tag C[s] This "s" in the tag field indicates that integration of new or same behavior is has taken place at the component C in state s.

13. Tag C[s] When a component C realizes an equivalent state s the behavior that operates is defined only by the state.

14. Tag C[s] When this component state is reached control needs to be passed to component C.

15. Tag C[s] As components of type C (two or more) are in state s, this is like the universal quantifier " s". We use "s" for "some or more"

16. Tag C[s] A percentage (some) or components of type C are in state s. This is like the existential quantifier " s".

17. Tag C[s] An arbitrary or irrelevant component of type C is in state s.
APPENDIX B
Usability Studies
INFORMATION PACK

Hello I am a PhD student studying usability design at Griffith University. I have already conducted two usability studies, but I need three people with Usability experience to help me analyze the think-aloud data, as a means of identifying usability Problems. The task of transcribing the user’s actions has already been done. What I need you to do is to go over this data, and highlight parts of the interaction you believe demonstrate a usability Problem. If the task proves overlong, or you are unable to finish due to the time to do these, contact me and I’ll try and sort out something.

There were two studies:

Study 1: VCR test (timer recording)
Six subjects were asked to set up a timer recording in several different ways across two different VCR models.

VCR Transcripts.doc

Study 2: Mobile Phone study.
Thirty subjects were asked to perform a range of tasks across two different mobile phone devices. There are a number of transcripts supplied:

30 – NEC Transcripts and 30 – Samsung Transcripts (30 subjects on two phones) in rtf format

Task sheets are supplied for both studies. These outline what the users had to do as part of the study. User manuals for both of the phones are supplied.

The size of the mobile phone transcripts will seem daunting. But much of the size is made up of key entry codes that you do not need to examine. Rely on the dialog and observation notes to find the areas that you believe to harbour a usability Problem. The next section documents how to record these usability Problems.

Page 451
How to Comment:

1. Open the document (e.g. VCR Transcripts doc) some of the files are in RTF format,


The best way to do this is to use the Reviewing Toolbar to insert comments. You can access the reviewing toolbar by Accessing the View Menu, selecting Toolbars, and then checking the review toolbar. The reviewing toolbar looks as follows.

![Reviewing Toolbar]

The only button you need to know is the insert comment button, as circled above.

3. Select the text you believe constitutes a usability Problem first. As you would any other piece of text.

An example of this is as follows:

**Zoe: test tape 1**

```
Zoe: Can I use the two? -- That one or this? Do I have to read instructions?
AI: Instructions are in the manual
Silence - pages turning
3.24: AI: the remotes not working now.
Zoe mumbles
AI: Yeah! No worries
4.11 Zoe: Channel one. Mmm. Oh, what's here?
```

After the text has been selected you can now insert a comment.

4. Insert a Comment

Press the Insert Comment Button on the Reviewing toolbar

![Insert Comment Button]

An enlarged picture of this button is shown here. It is usually on the far left of the reviewing toolbar.

When you click this button the bottom of the screen will change as follows (marked in the black rectangle on the last diagram)
Type the comment in here. LEAVE THE COMMENT box open

From this point on, you just need to repeat the following process.

1. Scroll down in main window to find text, which suggests a Problem
2. Select the block of text you believe suggests a Problem.
3. Press insert comment on Reviewing toolbar (text will now be highlighted in yellow.
4. Type comment in comment box
5. Repeat this process.

It's all fairly simple. You can move between comments using the comment window at the bottom of the screen, if you need to fix one already typed in. Other buttons on the Reviewing bar, can also be used to fix comments.

When you are finished SAVE THE FILE

Do this for both the VCR Transcripts file and the Mobile Phone transcripts.
How to Grade Usability Problems

The study has a specific way of grading problems. Since some Problems are more serious than others, the first measure is the severity of the Problem. When you highlight Problems and insert comments, include a code for the severity, and then appropriate codes for the attributes.

Severity

Note first the level of seriousness of the Problem. Just write a number between (1 and 4). These numbers correspond to 4 types of usability Problems as follows:

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Prevent Completion of a Task</td>
</tr>
<tr>
<td>2</td>
<td>Create significant delay and frustration</td>
</tr>
<tr>
<td>3</td>
<td>Problems have a minor effect on usability</td>
</tr>
<tr>
<td>4</td>
<td>Problems are more subtle and often point to an enhancement that can be added in the future.</td>
</tr>
</tbody>
</table>

Table 4.3 Severity of usability Problems, Dumas and Redish (1999:323-324).

Attributes

As part of my PhD, I have been looking at various attributes, believed to contribute to the eventual usability of a product. There are quite a few of these. I am more interested in nailing down the severity of a Problem. So don't let these codes slow you down too much. However, if you are able to note the codes for these in your comments it would be greatly appreciated. These attributes and their respective codes are as follows:

Forgiveness (FG)

Forgiveness is the acceptance and prevention of human error. This includes automatically formatting text when a number or sequence of characters is incorrect.
Appendix B

Error Recovery (ER)
Whether the system permits commands to be cancelled or reversed and for the user to return to a
given point if an error occurs (essentially error correction). Can a user undo their actions?

Responsiveness (RS)
Responsiveness is the minimisation of response time and the effective provision of feedback. Does
the system respond to the user's actions quickly, and does its response tell the user what is
happening?

Controllability (CT)
The user must 'control' the interaction. System Actions should always result from a user's request,
system actions must be able to be interrupted and stopped, and the user should never be
interrupted by system errors

Directness (DR)
Can the user see what needs to be done, and then observe the results, using the visual objects
provided by the interface?

Efficiency (EF)
These are the minimisation of the user's effort, and the efficient execution of system processes to
support the user's increased productivity.

Learnability (LN)
The ease of learning, the time to learn, the structure of information, and what learning support is
provided. How easily do the users take to learn the system, and perform the tasks.

Predictability (PR)
Predictability is a measure of how well the interface supports the logical flow of interaction.
According to Galitz, the "user should be able to anticipate the natural progression of each task".
Does the system support the user's ability to determine what to do next.

Simplicity (SM)
Simplicity is simply the reduction of complexity across all aspects of the interface. (e.g. only
necessary information is displayed).
Transparency (TN)

It is the concealment of the workings and technical details of the system to facilitate interaction. In a transparent user interface users should not need to be aware of the system mechanics in order to use the interface.

Familiarity (FM)

Familiarity is an interesting quality, since it relies on the user’s previous experience. Familiarity is closely tied to three factors:

4. Familiar concepts and language
5. A natural interface, which mimics the patterns of the users’ behaviour
6. ‘Real world metaphors (e.g the desktop)’ (Galitz, 1997: 41)

Consistency (CS)

The same action must have the same effect every time. And actions with the interface must be consistent with the performance other actions. E.g. Deleting a word must be consistent with the process for deleting a sentence

Configurability (CF)

Configurability is the degree to which the software allows the users to pick preferences and settings. The user may structure the environment to suit their needs and preferences. However caution must be taken that these settings are easily reversible and reset to default settings.

Flexibility (FL)

Flexibility is a system’s capability to respond to individual differences between users. Thus the provision of short cuts for expert users is such an example.

User Satisfaction (US)

Is the user satisfied with the system or are they constantly frustrated.

Visual Appeal and Layout (VA)

Infrequent (less used) controls are hidden, and those that are used more frequently are more obvious.
I do not want these attributes to slow you down too much. Keep a page available with all the codes, and do your best to assign them to usability problems. A summary for these codes is as follows. Contact me if you need more information.

<table>
<thead>
<tr>
<th>Forgiveness</th>
<th>FG</th>
<th>Prevent Problems</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error Recovery</td>
<td>ER</td>
<td>Fix Problems</td>
</tr>
<tr>
<td>Responsiveness</td>
<td>RS</td>
<td>Quick Response Feedback</td>
</tr>
<tr>
<td>Controllability</td>
<td>CT</td>
<td>User in control – Can cancel actions</td>
</tr>
<tr>
<td>Directness</td>
<td>DR</td>
<td>Direct actions and visual objects assist user</td>
</tr>
<tr>
<td>Efficiency</td>
<td>EF</td>
<td>Tasks can be done rapidly and effectively</td>
</tr>
<tr>
<td>Learnability</td>
<td>LN</td>
<td>Easy to learn, does system support learning</td>
</tr>
<tr>
<td>Predictability</td>
<td>PR</td>
<td>Users can predict what the user will do in response</td>
</tr>
<tr>
<td>Simplicity</td>
<td>SM</td>
<td>Minimization of complexity, the less needed is more hidden</td>
</tr>
<tr>
<td>Transparency</td>
<td>TN</td>
<td>User does not need to know inner workings of system</td>
</tr>
<tr>
<td>Familiarity</td>
<td>FM</td>
<td>Interface is familiar to other external interfaces</td>
</tr>
<tr>
<td>Consistency</td>
<td>CS</td>
<td>Actions within interface are consistent applied</td>
</tr>
<tr>
<td>Configurability</td>
<td>CF</td>
<td>User can change settings, and also reverse them</td>
</tr>
<tr>
<td>Flexibility</td>
<td>FL</td>
<td>Different techniques for experts and novices are available</td>
</tr>
<tr>
<td>User Satisfaction</td>
<td>US</td>
<td>The user is satisfied with task (not frustrated)</td>
</tr>
<tr>
<td>Visual Appeal and Layout</td>
<td>VA</td>
<td>Less used functions are hidden, more frequent ones are obvious.</td>
</tr>
</tbody>
</table>

Here’s a quick example. Text that is highlighted:

I couldn’t do it the first, which was bizarre. I was dumb. Which is Number 6 VCR TV thing? I’ll follow my own instincts. OK so it will be somewhere here OK. Oooh! Groovy!

Comment Box: (this is an example only) don’t let this influence your own codes
3 LN, US (this code means 2 for severity and codes for Learning and User Satisfaction.

The severity is more important than the attributes, so make sure it is never omitted in a comment.
# TIMER RECORDING STUDY - TASKS

<table>
<thead>
<tr>
<th>NV-HD620 MK2A with TV display</th>
<th>Done</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1:</td>
<td></td>
</tr>
<tr>
<td><strong>Open Manual: Pages 24 – 25 Timer Recording</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Program the following:</strong></td>
<td></td>
</tr>
<tr>
<td>1. (Saturday): Channel Nine: 10.00 a.m. – 12.00 p.m. Football</td>
<td></td>
</tr>
<tr>
<td>2. (Monday – Friday): ABC: 7.00 p.m – 7.30 p.m. News</td>
<td></td>
</tr>
<tr>
<td>3. (Every Sunday): ABC 12.00 p.m. – 1.00 p.m. Landline</td>
<td></td>
</tr>
<tr>
<td><strong>Check all programs against above</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Correct this program &lt;1&gt; Saturday Channel Nine 10.00 a.m. 12.30 p.m. Football</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Cancel All programs</strong></td>
<td></td>
</tr>
<tr>
<td>Task 2:</td>
<td></td>
</tr>
<tr>
<td><strong>Open Manual: Pages 26 – G Code Programming (Refer to TV Guide for G-Codes).</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Program the following:</strong></td>
<td></td>
</tr>
<tr>
<td>4. (Friday): ABC: 11.55p.m – 5.30a.m Rage</td>
<td></td>
</tr>
<tr>
<td>5. (Friday) Seven 11.00 a.m. – 11.30 a.m. Ricky Lake</td>
<td></td>
</tr>
<tr>
<td>6. (Friday-Saturday) SBS 9.00pm – 9.15 World News</td>
<td></td>
</tr>
<tr>
<td><strong>Check all programs against above</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Correct: &lt;3&gt; (Saturday) SBS 9.00 p.m. – 9.30 pm. World News</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Cancel All Programs</strong></td>
<td></td>
</tr>
</tbody>
</table>

These tasks varied according to what day, the subjects conducted the study. However the two sheets show the default tasks for the study. The Problems should be evident from the users talking aloud, and I will be assessing the variation in the tasks. So you shouldn’t be too concerned with these variations.
## TASKS for second VCR model

### VCR Model NV-SD20 (No TV Required)

<table>
<thead>
<tr>
<th>Task 1: Open Manual Pages 24 – 25 Timer Recording Set timer recording for the following</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4. (Saturday): Channel Nine: 10.00 a.m – 12.00 p.m. Football</td>
<td></td>
</tr>
<tr>
<td>5. (Monday – Friday): ABC: 7.00 p.m. – 7.30 p.m. News</td>
<td></td>
</tr>
<tr>
<td>6. (Every Sunday): ABC 12.00 p.m. – 1.00 p.m. Landline</td>
<td></td>
</tr>
<tr>
<td>Check Programs Correct: Correct: &lt;1&gt; Saturday Channel Nine 10.00 a.m. 12.30 p.m. Football Cancel All Programs</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Task 2: Open Manual Pages 26 Using the Remote Controller Set timer recording for the following</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4. (Thursday): Channel Nine: 11.30p.m – 12.45 a.m Twilight Zone</td>
<td></td>
</tr>
<tr>
<td>5. (Thursday-Friday): Channel Seven: 11.00 a.m. – 11.30 a.m. Ricky Lake</td>
<td></td>
</tr>
<tr>
<td>6. (Every Saturday) SBS 1.00 pm – 2.00 p.m. Journal</td>
<td></td>
</tr>
<tr>
<td>Check Correct &lt;1&gt; (Thursday): Channel Nine: 11.30p.m – 12.30 a.m. Twilight Zone Cancel</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Task 3: Open Manual: Pages 34 – 35 Channel Nine: 11.30p.m – 12.30a.m Twilight Zone Set timer recording for the following</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4. (Thursday): SBS 9.00p.m – 9.15pm World News</td>
<td></td>
</tr>
<tr>
<td>5 (Thursday - Friday): Channel Seven: 4.00p.m- 4.30 p.m. Big Arvo</td>
<td></td>
</tr>
<tr>
<td>6. (Every Thursday) Nine 9.30 a.m – 11.00 a.m. Mornings with Kerri-Anne</td>
<td></td>
</tr>
<tr>
<td>Check Correct: &lt;1&gt; (Thursday): SBS 9.00p.m – 9.30pm World News Cancel All Programs</td>
<td></td>
</tr>
</tbody>
</table>
TASK SHEETS for Mobile Study

These are a list of the tasks conducted by subjects in the mobile phone study.

MOBILE PHONE STUDY PART [Samsung]

This is a usability study about Mobile Phone Usage. The following tasks involve a number of different functions across two different phones and networks. This is to assess the design of the NEC, and not your performance as a user. The videotaping will be used to determine the difficulty of these operations.

Please complete the following tasks. You may tick tasks in the right column as they are completed. This task is 40 minutes long. You are encouraged to think aloud. You should spend half a minute trying to complete each task without accessing the manual. The Ref column represents the page number in the manual that corresponds to each of the tasks. You are encouraged to complete each task as quickly as you can.

<table>
<thead>
<tr>
<th>#</th>
<th>Task Description</th>
<th>Ref</th>
<th>Exampl</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Turn phone on</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Make call to 0407 515 530</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Add following names and number to phone book (contacts)</td>
<td>35-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Name: Al, Number: 0407 515 530 to phone book (contacts)</td>
<td>36</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Name: Al, Number: 3848 2885 to phone book (contacts)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Activate Call history – find 0407 515 530 and return call using call history</td>
<td>55</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Access Web Browser</td>
<td>113</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Use Speed Dial – Al (0407 515 530)</td>
<td>39</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Create SMS: with “Hello” as message. Save message</td>
<td>61</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Use calculator add (5 + 2)</td>
<td>103</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Assign Voice Recording to Al (phone book record)</td>
<td>67-</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>68</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Create appointment in Scheduler for October 19 2004 2.30 p.m lunch.</td>
<td>99</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Use Voice Dial to call – [Al]</td>
<td>67,68</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Delete appointment in Scheduler for October 19 2004: 230 p.m</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Send saved SMS “hello” to Al</td>
<td>36-</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>38</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Delete Al from phone Book (contents)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Turn Phone off</td>
<td>20</td>
<td></td>
</tr>
</tbody>
</table>
# MOBILE PHONE STUDY PART [NEC]

<table>
<thead>
<tr>
<th>#</th>
<th>Using NEC e650i</th>
<th>Ref</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Turn phone on</td>
<td>3.3</td>
</tr>
<tr>
<td>2</td>
<td>Make call to 0407 515 530</td>
<td>4.1</td>
</tr>
</tbody>
</table>
| 3  | Add following names and number to phone book (contacts)  
   Name: A1, Number: 0407 515 530 to phone book (contacts)  
   Name: A1, Number: 3848 2885 to phone book (contacts) | 5.4 |
| 4  | Activate Call history – find 0407 515 530 and return call using call history | 4.4 |
| 5  | Access 3 Web Services | 2.1 |
| 6  | Navigate Web services to find Java game: PlayGolf and download it. | 6.1, 6.12 |
| 7  | Take 12 sec Video Sequence – Cancel Save | 7.6 |
| 8  | Use calculator add (5 + 2) | 11.8 |
| 9  | Take photo of glass and save | 7.2 |
| 10 | Create appointment in calendar for October 19 2004 2:30 p.m lunch. | 11.3 |
| 11 | Create and send email  
   To: Alistair@fusemail.com  
   Subject: Test  
   Content: Hello | 8.3 |
| 12 | Create SMS message with “Hello” - Save it to memory | 9.2 |
| 13 | Delete appointment in calendar for October 19 2004: 230 p.m | 11.5 |
| 14 | Send stored photo to 0412 3848 10: | 9.4 |
| 15 | Activate Java Game: Begin game: PlayGolf: Close Java Game | 6.13 |
| 16 | Delete all “A1”s from phone Book (contacts) | 5.6 |
| 17 | Delete stored Photo | 7.5 |
| 18 | Delete SMS Message | 7.6 |
| 19 | Turn Phone off | 3.3 |

Thank you for helping out. Please try and analyse this data as soon as possible. The video study analysis should not take too long as it is relatively small and in a single file. The Mobile phone study is split into sixty smaller files. I would encourage you to do it as rapidly as you can, without missing the most obvious and important Problems.
DIAGRAMS of DEVICES – to help understand tasks

VCR: SD20

Check Prog  Timer Record
Minus or Down  Check Prog
Plus or Up  Next

Speed  Start Time (H:M)
Channel  Day
Date  End Time (H:M)  Program
APPENDIX C
Kieras’s VCR NGOMSL Example

Example: VCR Procedures

User goals - a subset
• Set the clock
• Record a program
• Stop current recording
• Cancel previously programmed recording

Assumptions
• A cassette is loaded
• Power is on
• Desired channel is already selected

Methods and Selection Rules are somewhat simplified

Example uses NGOMSL notation.
Methods for Setting the Clock

Method for goal: set the clock
Step 1. Press CLOCK button
Step 2. Accomplish goal: set a day and time with the current
day and time
Step 3. Press CLOCK button
Step 4. Return with goal accomplished

Method for goal: set a day and time
Step 1. Accomplish goal: set day value to the desired value
Step 2. Press SELECT
Step 3. Accomplish goal: set hour value to the desired value
Step 4. Press SELECT
Step 5. Accomplish goal: set minute value to the desired
value
Step 6. Return with goal accomplished

Method for goal: set a value to a desired value
Step 1. Verify that display shows current value is flashing
Step 2. Decide: If display shows value same as desired,
then return with goal accomplished.
Step 3. Decide: If display shows value smaller than desired,
then press UP button.
else press DOWN button.
Step 4. Go to Step 2.
Methods for Recording a Program

Selection rule set for goal: record a program
If you are present when the program starts
and you will be present when the program ends
then accomplish goal: record a program manually
If you are present when the program starts
and you will not be present when the program ends
and you know how long the program lasts
and the VCR clock is set
then accomplish goal: record a program with One-Touch Recording
If you will not be present when the program starts
and you know when the program will start
and you know how long the program lasts
and the VCR clock is set
then accomplish goal: record a program with Timer Recording
Return with goal accomplished

Method for goal: record a program manually
Step 1. Wait for program to start.
Step 2. Hold down REC button.
Step 3. Press PLAY button.
Step 4. Release both buttons.
Step 5. Verify that display shows "REC" and arrow is moving
Step 6. Wait for program to end
Step 7. Press STOP button.
Step 8. Return with goal accomplished.

Method for goal: record a program with One-Touch Recording
Step 1. Wait for program to start
Step 2. Press OTR button.
Step 3. Press OTR button.
Step 4. Decide: If time shown in display is less than length of program, go to Step 3.
Step 5. Return with goal accomplished.
Methods for Recording a Program (continued)

Method for goal: record a program with Timer Recording
Step 1. Press PROGRAM button.
Step 2. Verify that program number flashes
Step 3. Press SELECT button.
Step 4. Accomplish goal: set a day and time with the starting day and time of the program
Step 5. Press SELECT button
Step 6. Accomplish goal: select recording length
Step 7. Press SELECT button
Step 8. Accomplish goal: set channel value to desired value
Step 9. Press CLOCK button
Step 10. Verify the display returns to normal display
Step 11. Press POWER button
Step 12. Verify that display shows "TIMER" Step 13. Return with goal accomplished

Method for goal: select the recording length
Only certain values are allowed, so special method required
Step 1. Verify that display shows "LGTH" Step 2. Press UP button.
Step 3. Decide: If time shown in display is less than length of program, go to Step 2.
Step 4. Return with goal accomplished.
Methods for Stopping and Canceling a Recording

Selection rule set for goal: stop an on-going recording
There are subtle cues on the display that could also be
used to select the method
If you remember that the recording is manual
then accomplish goal: stop an on-going manual recording
If you remember that the recording is a One-Touch
Recording then accomplish goal: stop a One-Touch
Recording
If you remember that the recording is a Timer
Recording then accomplish goal: stop a Timer
Recording
Return with goal accomplished.

Method for goal: stop an on-going manual recording
Step 1. Press the STOP button
Step 2. Verify that "REC" disappears and arrow stops
moving. Step 3. Return with goal accomplished.

Method for goal: stop a One-Touch Recording
Actually behaves differently depending on time delay since
OTR button last pressed, but this method always works
Step 1. Press OTR button.
Step 2. If display shows a time greater than zero, go to Step 1.
Step 3. Wait several seconds
Step 4. Verify that VCR turns off.
Step 5. Return with goal accomplished.
Methods for Stopping and Canceling a Recording (continued)

Method for goal: stop a Timer Recording
Step 1. Press POWER
Step 2. Verify that display shows "REC" disappears and arrow stops moving.
Step 3. Accomplish goal: cancel a Timer Recording
Step 4. Press POWER
Step 5. Verify that VCR turns off.
Step 6. Return with goal accomplished.

Method for goal: cancel a Timer Recording
Step 1. Press PROGRAM
Step 2. Press SELECT
Step 3. If Display does not show "LENGTH" go to Step 2.
Step 4. Press DOWN.
Step 5. If Display does not show zero, go to Step 4.
Step 6. Press CLOCK
Step 7. Return with goal accomplished
APPENDIX D

GSE Step 3 for NGOMSL VCR Model

Incrementally Integrate Individual Requirements Behaviour Trees to Create an Evolving Design Behaviour Tree.

The third stage of the GSE procedure involves putting the individual requirements trees together. Dromey (2002b) describes the integration of these trees as building a ‘jigsaw puzzle’. He strongly encourages adding pre-conditions, as either a new root node or sub tree, since these conditions can prove a critical step to linking requirements into a comprehensive model.

This process begins with an RIC or Root Integration table. In accordance with Dromey (2002a) the REQ column corresponds to the requirement number, the Type to the type of component (system S or component C), the root node column for the root node of the specified requirement, and the “occurs in” column corresponds to any requirement where this root node exists in the behaviour tree. The fig. Column corresponds to the diagram of a given behaviour tree in this chapter. Finally, since this is a GOMS Formatted Description, the GOMS column specifies whether the behaviour tree corresponding to a requirement is a selection rule (S) or method (M). Since methods contain a set of operators, and methods correspond to goals, additional syntax is not needed. In keeping with the behaviour trees, common assumptions are drafted as R0. Any requirement whose root node does not exist in any other requirement (expect for the assumptions themselves) is specified as being linked to the assumptions.

<table>
<thead>
<tr>
<th>Fig</th>
<th>Req</th>
<th>Root Cpt State</th>
<th>Type</th>
<th>GOMS</th>
<th>Root Node</th>
<th>Occ</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.2</td>
<td>0</td>
<td></td>
<td>C</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.3</td>
<td>1</td>
<td>Set the Clock</td>
<td>C</td>
<td>M</td>
<td>Goal [Set the Clock]</td>
<td>R0</td>
</tr>
<tr>
<td>5.4</td>
<td>2</td>
<td>Set a Day and Time</td>
<td>C</td>
<td>M</td>
<td>Goal [Set a Day and Time]</td>
<td>R1</td>
</tr>
<tr>
<td>5.5</td>
<td>3</td>
<td>Set a Value to a Value</td>
<td>C</td>
<td>M</td>
<td>Goal [Set a Value to a Value]</td>
<td>R2</td>
</tr>
<tr>
<td>5.6</td>
<td>4</td>
<td>Record a Program</td>
<td>C</td>
<td>S</td>
<td>Goal [Record a Program]</td>
<td>A</td>
</tr>
<tr>
<td>5.7</td>
<td>5</td>
<td>Manually</td>
<td>C</td>
<td>M</td>
<td>Goal [Record a program manually]</td>
<td>R4</td>
</tr>
<tr>
<td>5.8</td>
<td>6</td>
<td>One Touch</td>
<td>C</td>
<td>M</td>
<td>Goal [Record a program with OTR]</td>
<td>R4</td>
</tr>
<tr>
<td>5.9</td>
<td>7</td>
<td>Timer Recording</td>
<td>C</td>
<td>M</td>
<td>Goal [Record a program with Timer]</td>
<td>R4</td>
</tr>
<tr>
<td>5.10</td>
<td>8</td>
<td>Set the Recording Length</td>
<td>C</td>
<td>M</td>
<td>Goal [Set the Recording Length]</td>
<td>R7</td>
</tr>
<tr>
<td>5.11</td>
<td>9</td>
<td>Stopping an Ongoing Recording</td>
<td>C</td>
<td>S</td>
<td>Goal [Stop an ongoing recording]</td>
<td></td>
</tr>
<tr>
<td>5.12</td>
<td>10</td>
<td>Stop an ongoing manual recording</td>
<td>C</td>
<td>M</td>
<td>Goal [Stop an ongoing manual recording]</td>
<td>R9</td>
</tr>
<tr>
<td>5.13</td>
<td>11</td>
<td>Stop a one touch recording</td>
<td>C</td>
<td>M</td>
<td></td>
<td>R10</td>
</tr>
<tr>
<td>5.14</td>
<td>12</td>
<td>Stop a timer recording</td>
<td>C</td>
<td>M</td>
<td></td>
<td>R11</td>
</tr>
<tr>
<td>5.15</td>
<td>13</td>
<td>Cancel a Timer Recording</td>
<td>C</td>
<td>M</td>
<td></td>
<td>R12</td>
</tr>
</tbody>
</table>

Requirements Interaction Table (RIC)
Linking R0: Assumptions and R1: Set the Clock to form Design 0,1

When joining the behaviour trees together there are four possible scenarios that may occur:

1. **Asymmetric integration**
   A root node for one tree exists in another.

2. **Symmetric integration**
   Both trees contain root nodes corresponding to each other. Either tree may be used.

3. **Deficiency**
   There are no root matches in either tree. The RIIC should be altered, for the relevant requirement to ensure asymmetric or symmetric integration.

4. **Overlap and Redundancy**
   Where this occurs, redundancy is discarded.

Each of these conditions is a possibility when linking the behaviour trees to one another. Starting with the first two behaviour trees, the assumptions and goal number 1, each behaviour tree will be integrated in accordance with the above conditions.

In requirement 0, three assumptions were listed: the VCR should be on, a tape should be inserted and a channel must be set to the desire channel. From a logical perspective it is hard to see why a tape “must be inserted” before the clock can be set. Since no tape is played, stopped or manipulated, it seems an illogical precondition for setting the clock. However for the time being at least these original assumptions are considered to be valid.

Looking at the NIIC table, the root node of R1 does not actually occur in any of the behaviour trees listed in the NIIC table. Therefore, in keeping with the model it must be linked directly to the assumptions that Kieras specifies for this model. Joining these two together is a simple matter of joining the two at the top of the R1 behaviour tree. This process is demonstrated in the following diagram (over the page).
The linkage between the assumptions and the first behaviour tree provides a simple asymmetric connection, where the three assumptions are used as the root node. There are other nodes, Goals 4, and 6, which also lack any root node connection. They will also be linked to the assumptions in much the same way, later in this section.

When these behaviour trees are linked together, they become evolving design behaviour trees. In order to indicate this transformation, the Requirement 0 (R0) and Requirement 1 (R0) trees become a new Design behaviour tree (represented as D) for R0 and R1. This new design tree is denoted as D0-1. This is in accordance with GSE conventions specified by Dromey (2002a).

Rules for Linking the Behaviour Trees

While it may seem logical to link, R2 (set a day and time) to R1 closer examination of the method makes this an unlikely proposition. R2 is a method called by two other methods, and is therefore deemed to be global. The same applies to R3, which is a generic method for setting a value to a value. These are called as encapsulated processes using the @ symbol by any method that calls them.
In order to unite the models, several rules were established based on the unique requirements for uniting a NGOMSL model into a single GSE entity. These rules are as follows:

1. When a change of state at the end of a sub method, is superseded by the same component in the calling method, the calling state takes precedence.

2. When a sub method is integrated into its parent node, the “return with goal-accomplished” nodes are not carried over.

3. Where a sub method is called as using an encapsulated node, its frequency and dependence on the calling method must be used to determine whether it will be pasted into the tree. This leads to two possibilities:
   a. Where the same components in a sub method are called by the same name in the tree containing the encapsulation, and the sub method is called once only, and then this goal is a logical continuation of the parent and may be joined
   b. Where the encapsulation, uses generic descriptions of components, and appears multiple times, this is an independent tree, and not to be linked to the parent. Instead it should be specified along side the other behaviour trees, once only.
   c. When a sub method is called a single time from an encapsulated node across all trees, yet uses generic descriptions of components it is at the author’s discretion as to whether options a or b will be followed. However, if the sub method is contains generic components, which are not equivalent to those in the parent, it is recommended that the tree remain a separate entity as in step b. The reason for this is to preserve the integrity of the tree and acknowledgement that later extensions to the behaviour of the model.

**Procedure followed to build the trees.**

Based on the proceeding the following steps can be used to build a single overall GSE model

1. Requirement 2 is an independent process. (Rule 3b)
2. Requirement 3 is an independent process (Rule 3b)
3. Requirement 4 can be linked to Design 0,1 to make Design 0,1,4 (Rule 3a)
4. Requirement 5 can be linked to Design 0,1,4 to make Design 0,1,4,5
   (Rule 3a)
5. Requirement 6 to Design 0,1,4,5 to make Design 0,1,4,5,6
6. Requirement 7 to Design 0,1,4,5,6 to make Design 0,1,4,5,6,7
7. Requirement 8 to Design 0,1,4,5,6,7 to make Design 0,1,4,5,7
APPENDIX E
Mobile Phone - Result Tables

These tables contain potential problem instances, for the mobile phone study, for each of the tasks on each phone, compiled using the examiners’ comments (see section 7.3.1 for more information).

### NEC Task Tables.

<table>
<thead>
<tr>
<th>Inst</th>
<th>Subj</th>
<th>Comments</th>
<th>Severities and Attributes</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B</td>
<td>D1</td>
<td>2,DR</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>M</td>
<td>S1</td>
<td>2,FM,RS</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>B</td>
<td>A1 S1 D2</td>
<td>3,FM, LN, 3, DR, 2, DR</td>
<td>2.67</td>
</tr>
<tr>
<td>4</td>
<td>M</td>
<td>A1 D1</td>
<td>3,EF, CT, 3, US, DR</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>T</td>
<td>A1[D1 S1]</td>
<td>3, DR, CT, FM, LN[3, DR, 3, DR]</td>
<td>3</td>
</tr>
</tbody>
</table>

Examiner Results for NEC Task 1: Turning the Phone On

<table>
<thead>
<tr>
<th>Inst</th>
<th>Subj</th>
<th>Comments</th>
<th>Severities and Attributes</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>T</td>
<td>S2</td>
<td>3, RS, ?</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>A2 [D3</td>
<td>4, PR, [4, DR], [3, FM,</td>
<td>3.67</td>
</tr>
</tbody>
</table>

Examiner Data for NEC Task 2: Making a Call

<table>
<thead>
<tr>
<th>Inst</th>
<th>Subj</th>
<th>Comments</th>
<th>Severity and Attributes</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>K</td>
<td>D3 A2 S2</td>
<td>1, US, DR, 1, DR, US, PR, FM, 1, PR, DR,</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>E</td>
<td>A2 D2 S2</td>
<td>2, DR, VA, LN, FM, 2, DR, US, 2, ER, CT</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>B</td>
<td>D5 S4</td>
<td>2, DR, SM, 2, DR,</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>B</td>
<td>A7 D10</td>
<td>2, US, 2, US,</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>B</td>
<td>A8</td>
<td>2, LN, PR, DR,</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>M</td>
<td>S4</td>
<td>2, RS, SM, ?</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>T</td>
<td>D4</td>
<td>2, DR,</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>T</td>
<td>S6</td>
<td>2, RS,</td>
<td>2</td>
</tr>
<tr>
<td>9</td>
<td>B</td>
<td>A5 D8 S7</td>
<td>2, VA, US, LN, SM, DR, FM, 3, US, DR, 2, PR, FM,</td>
<td>2.33</td>
</tr>
<tr>
<td>11</td>
<td>B</td>
<td>A6 D9</td>
<td>3, US, 2, US,</td>
<td>2.5</td>
</tr>
<tr>
<td>12</td>
<td>E</td>
<td>A1 D1 S1</td>
<td>3, DR, 3, DR, PR, 2, DR,</td>
<td>2.67</td>
</tr>
<tr>
<td>13</td>
<td>K</td>
<td>A1 D2 S1</td>
<td>3, PR, SM, 3, DR, 2, PR,</td>
<td>2.67</td>
</tr>
<tr>
<td>14</td>
<td>B</td>
<td>D4 [S3]</td>
<td>3, DR, PR[3, DR]</td>
<td>3</td>
</tr>
<tr>
<td>15</td>
<td>B</td>
<td>D6 [A3[S5]]</td>
<td>3, DR, PR, [3, PR, DR, VA, RS, [3, PR, CT, ]</td>
<td>3</td>
</tr>
<tr>
<td>16</td>
<td>B</td>
<td>A4 D7 S6</td>
<td>3, DR, CT, PR, US, LN, RS, FM, 3, PR, US, 3, CT,</td>
<td>3</td>
</tr>
</tbody>
</table>
Examiner Data for NEC Task 3: Adding a Contact

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>K</td>
<td>A3 D5</td>
<td>2, TN, SM, 2, TN, DR</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>M</td>
<td>D6</td>
<td>2, PR, DR</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>T</td>
<td>D6 S7 A5 S8</td>
<td>2, DR, [2, DR], [2, DR, LN, FM], [2, DR], 2, DR, VA</td>
<td>2.33</td>
</tr>
<tr>
<td>4</td>
<td>E</td>
<td>D3 A3 S3</td>
<td>1, DR, PR, FM, CS, [3, DR], 3, DR, EF</td>
<td>2.33</td>
</tr>
<tr>
<td>5</td>
<td>B</td>
<td>A9 S8 D17 D18</td>
<td>2, PR, DR, FG, CT, RS, [3, DR], 3, US, DR, 2, ER, DR</td>
<td>2.5</td>
</tr>
<tr>
<td>6</td>
<td>B</td>
<td>D16</td>
<td>3, PR</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>B</td>
<td>D19</td>
<td>3, US, PR, FM</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>M</td>
<td>D5</td>
<td>3, PR</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>M</td>
<td>D7</td>
<td>3, PR</td>
<td>3</td>
</tr>
<tr>
<td>10</td>
<td>R</td>
<td>A3 D3 S3</td>
<td>3, RS, 3, DR, 3, LN, ?, DR, ?</td>
<td>3</td>
</tr>
<tr>
<td>11</td>
<td>R</td>
<td>D4</td>
<td>3, DR</td>
<td>3</td>
</tr>
</tbody>
</table>

Examiner Data for NEC Task 4: Activate Call History
<table>
<thead>
<tr>
<th>Inst</th>
<th>Subj</th>
<th>Comments</th>
<th>Attributes and Severity</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>K</td>
<td>D7</td>
<td>2,DR,</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>K</td>
<td>S3</td>
<td>2,FM,</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>M</td>
<td>A6</td>
<td>2,EF,TN,</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>M</td>
<td>D8</td>
<td>2,PR,DR,</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>R</td>
<td>A4 D5 S4</td>
<td>2,ER,LN,CT,DR, 3,DR,PR,3,DR,?</td>
<td>2.67</td>
</tr>
<tr>
<td>6</td>
<td>T</td>
<td>[D7[S9][S10]] - altered</td>
<td>[3 DR,PR,FL, [3,DR,FM,][3,VA,SM,]][2, TN,RS, [3,VA,SM,]]</td>
<td>2.8</td>
</tr>
<tr>
<td>7</td>
<td>T</td>
<td>A6[D8] - altered</td>
<td>3,DR</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>E</td>
<td>S5</td>
<td>3,RS,</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>B</td>
<td>A10 D20 S9</td>
<td>3,RS, 3,DR, 3,RS,</td>
<td>3</td>
</tr>
<tr>
<td>11</td>
<td>K</td>
<td>D6</td>
<td>3,DR,</td>
<td>3</td>
</tr>
<tr>
<td>12</td>
<td>R</td>
<td>A5</td>
<td>3, TN,DR,CT,</td>
<td>3</td>
</tr>
</tbody>
</table>

Examiner Data for NEC Task 5 Accessing the '3' Web Services

<table>
<thead>
<tr>
<th>Inst</th>
<th>Subj</th>
<th>Comments</th>
<th>Severities and Attributes</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>M</td>
<td>D9[A7]</td>
<td>2,DR,PR[2,TN,]</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>E</td>
<td>A4</td>
<td>3,ER,TN,</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>K</td>
<td>A4</td>
<td>3,EF,</td>
<td>3</td>
</tr>
</tbody>
</table>

Examiner Data for Task 6: Play Game

<table>
<thead>
<tr>
<th>Inst</th>
<th>Subj</th>
<th>Comments</th>
<th>Severities and Attributes</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B</td>
<td>A14 D24</td>
<td>2,DR,FG,PR,CT, 2,US,DR,FG</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>D27[S14] [S15 A16]</td>
<td>2,US,DR,PR][3,DR,] [2,LN, 2,DR,PR,CT,LN]</td>
<td>2.25</td>
</tr>
<tr>
<td>3</td>
<td>B</td>
<td>A15 D25</td>
<td>2,DR,FM,PR,CT, 3,US,DR,CT</td>
<td>2.5</td>
</tr>
<tr>
<td>4</td>
<td>B</td>
<td>D26[S12]</td>
<td>3,FG,CS,[2,PR]</td>
<td>2.5</td>
</tr>
<tr>
<td>5</td>
<td>B</td>
<td>D30[S17]</td>
<td>3,DR[2,RS,]</td>
<td>2.5</td>
</tr>
<tr>
<td>6</td>
<td>M</td>
<td>D10[S7]</td>
<td>2,DR,PR[3,DR]</td>
<td>2.5</td>
</tr>
<tr>
<td>7</td>
<td>M</td>
<td>D11 S8</td>
<td>3,DR, 2,DR,CT,TN</td>
<td>2.5</td>
</tr>
<tr>
<td>9</td>
<td>B</td>
<td>S13</td>
<td>3,DR, ?,VA,?</td>
<td>3</td>
</tr>
<tr>
<td>10</td>
<td>B</td>
<td>A18 D29</td>
<td>3,US,PR,ER, 3,DR,</td>
<td>3</td>
</tr>
<tr>
<td>11</td>
<td>R</td>
<td>A6 S5</td>
<td>3DR, 3,DR,?</td>
<td>3</td>
</tr>
<tr>
<td>12</td>
<td>R</td>
<td>D6</td>
<td>3,DR,</td>
<td>3</td>
</tr>
</tbody>
</table>
### Examiner Data for NEC Task 7: Take a video sequence

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>R</td>
<td>D7</td>
<td>??</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>A21 [D33 S22]</td>
<td>3,PR,EF,DR,FM, [3,US, 2,DR,FM,]</td>
<td>2.67</td>
</tr>
<tr>
<td>3</td>
<td>B</td>
<td>D31[[S18][S19 A19][A20 S20]]</td>
<td>2,US,DR,PR[3,DR][3,DR,FM, 3,FM,DR,VA], [3,FM, 3,FM,]</td>
<td>2.83</td>
</tr>
<tr>
<td>4</td>
<td>B</td>
<td>D32 S21</td>
<td>3,US, 3,FM,</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>K</td>
<td>S4</td>
<td>3,RS,?</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>M</td>
<td>S9</td>
<td>4,DR,</td>
<td>4</td>
</tr>
</tbody>
</table>

### Examiner Data for NEC Task 8: Use Calculator

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>K</td>
<td>S5 A5 D8</td>
<td>4,RS, 4,DR,</td>
<td>4</td>
</tr>
</tbody>
</table>

### Examiner Data for NEC Task 9: Take a Photo and Save

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>M</td>
<td>A10 D17[S16]</td>
<td>1,DR,PR,CT,EF,TN,RS, 1,FG,DR[2,DR]</td>
<td>1.33</td>
</tr>
<tr>
<td>2</td>
<td>R</td>
<td>D10[S7]</td>
<td>1,DR,PR[2?]</td>
<td>1.5</td>
</tr>
<tr>
<td>3</td>
<td>B</td>
<td>S24</td>
<td>2,DR,</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>K</td>
<td>A6 D9 S6</td>
<td>2,LN,FM,DR, 2,DR,US, 2,DR,</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>M</td>
<td>S14</td>
<td>2,DR?</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>B</td>
<td>A25 D38 S26</td>
<td>3,RS, 3,DR,FG, 2,DR,</td>
<td>2.67</td>
</tr>
<tr>
<td>8</td>
<td>R</td>
<td>A7[D9 S6]</td>
<td>2,DR,PR,ER[3,DR,US, 3,DR]</td>
<td>2.67</td>
</tr>
<tr>
<td>10</td>
<td>E</td>
<td>S6[D5]</td>
<td>3,DR[3,DR]</td>
<td>3</td>
</tr>
<tr>
<td>12</td>
<td>B</td>
<td>D35</td>
<td>3,DR,</td>
<td>3</td>
</tr>
<tr>
<td>13</td>
<td>B</td>
<td>A22 D36</td>
<td>3,CT,DR, 3,DR,</td>
<td>3</td>
</tr>
<tr>
<td>15</td>
<td>M</td>
<td>D14[S12]</td>
<td>3,DR[3,DR,RS,]</td>
<td>3</td>
</tr>
<tr>
<td>16</td>
<td>M</td>
<td>A8 D15 S13</td>
<td>3,EF,DR,FM,FG,PR, 3,FG,3,FM,?,LN,?,DR,?</td>
<td>3</td>
</tr>
<tr>
<td>17</td>
<td>M</td>
<td>A9 D16</td>
<td>3,RS, 3,DR,US,</td>
<td>3</td>
</tr>
<tr>
<td>18</td>
<td>M</td>
<td>S15</td>
<td>3,RS,</td>
<td>3</td>
</tr>
<tr>
<td>19</td>
<td>R</td>
<td>D8</td>
<td>3,DR,</td>
<td>3</td>
</tr>
<tr>
<td>20</td>
<td>T</td>
<td>D10[S12]</td>
<td>3,DR[3,DR]</td>
<td>3</td>
</tr>
<tr>
<td>21</td>
<td>E</td>
<td>D6 A6</td>
<td>4,PR,FG,DR, 4,IN,RS,</td>
<td>4</td>
</tr>
</tbody>
</table>

### Examiner Data for NEC Task 10: Create an Appointment in Calendar
### Examiner Data for NEC Task 11: Compose and Send Email

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>M</td>
<td>S17</td>
<td>2,FM, LN, DR</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>D39[S27][S28]</td>
<td>2,DR, TN, [(2,DR)][3,FM,]</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>E</td>
<td>D7 S7</td>
<td>3, DR, PR, 2, DR</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>B</td>
<td>D42[A27]</td>
<td>2, US, LN, DR[3, DR, CT]</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>K</td>
<td>D10[A7][S7][S8][S9]</td>
<td>2, DR, PR, CS, [(2, DR, PR, FM, [3, FM, DR]), [3, FM, ] [3]?]</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>E</td>
<td>D8</td>
<td>3, DR</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>E</td>
<td>A7 D9 D10 [S8]</td>
<td>3, PR, FM, CS, LN, 3, US, FM, 3, VA, [3, FM,]</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>E</td>
<td>A8[D11 S9]</td>
<td>3, RS, VA, SM, [3, DR, 3, RS,]</td>
<td>9</td>
</tr>
<tr>
<td>10</td>
<td>B</td>
<td>A26[D40] [D41]</td>
<td>3, RS, [(3, PR), [3, US]</td>
<td>10</td>
</tr>
<tr>
<td>11</td>
<td>B</td>
<td>D43 : A28</td>
<td>3, RS, : 3, FM,</td>
<td>11</td>
</tr>
<tr>
<td>12</td>
<td>T</td>
<td>A7</td>
<td>3, DR, PR</td>
<td>12</td>
</tr>
<tr>
<td>14</td>
<td>T</td>
<td>S16</td>
<td>3, DR</td>
<td>14</td>
</tr>
<tr>
<td>15</td>
<td>E</td>
<td>A9 S10 D12</td>
<td>4, FM, 4, DR, 3, DR</td>
<td>15</td>
</tr>
</tbody>
</table>

### Examiner Data for NEC Task 12: Create an SMS and Save it

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B</td>
<td>D44[S29]</td>
<td>2,US, FG, PR[(3,DR)]</td>
<td>2.5</td>
</tr>
<tr>
<td>2</td>
<td>R</td>
<td>[A8[S8]] : D11</td>
<td>[3,RS,[3,DR]], ; 2,FG,DR,</td>
<td>2.67</td>
</tr>
<tr>
<td>3</td>
<td>E</td>
<td>A10[D13 S11]</td>
<td>3,PR, RS [4,DR 2, PR, RS,]</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>B</td>
<td>A29[D45]</td>
<td>3, PR, FG, EF, CT, RS, LN[3,DR]</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>B</td>
<td>D46</td>
<td>3, PR</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>M</td>
<td>A11[D18 S18]</td>
<td>3, RS,[3, RS, 3, RS,]</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>R</td>
<td>D12</td>
<td>3, DR, PR</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>R</td>
<td>S9</td>
<td>3?</td>
<td>3</td>
</tr>
</tbody>
</table>

### Examiner Data for NEC Task 14: Delete Appointment

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B</td>
<td>D48</td>
<td>2, DR</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>T</td>
<td>D13 A8</td>
<td>2, DR, 3, EF, PR</td>
<td>2.5</td>
</tr>
<tr>
<td>3</td>
<td>B</td>
<td>A31 D49 S30</td>
<td>2,RS, PR, 3, DR, PR, 3, RS,</td>
<td>2.67</td>
</tr>
<tr>
<td>5</td>
<td>M</td>
<td>A12</td>
<td>3, FG</td>
<td>3</td>
</tr>
</tbody>
</table>
## Appendix E

### Examiner Data for NEC Task 14: Send a stored Photo

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B</td>
<td>A16 [D22 S22]</td>
<td>1,SM,DR,VA,CS,FM, [2,DR,PR, 1,DR]</td>
<td>1.33</td>
</tr>
<tr>
<td>3</td>
<td>R</td>
<td>S12</td>
<td>2,DR,</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>R</td>
<td>S13</td>
<td>2,DR,?, Lost</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>R</td>
<td>D15 S14</td>
<td>2,DR, 2,DR,</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>K</td>
<td>D12 A8 S11</td>
<td>3,DR, 2,RS, 2,RS,?</td>
<td>2.33</td>
</tr>
<tr>
<td>7</td>
<td>E</td>
<td>A11[12]:[S13 D14]</td>
<td>2,EF,DR[2,DR]:[3,DR,RS, 3,EF,DR]</td>
<td>2.5</td>
</tr>
<tr>
<td>8</td>
<td>R</td>
<td>A9 S11 D13</td>
<td>2,RS,VA, DR, 3,DR, 2,DR,PR</td>
<td>2.5</td>
</tr>
<tr>
<td>9</td>
<td>M</td>
<td>A14 D20 S20</td>
<td>3,PR,FM,US, 3,US,PR, 2,FM</td>
<td>2.67</td>
</tr>
<tr>
<td>10</td>
<td>M</td>
<td>D21 [A15 S21]</td>
<td>2,DR, [3,PR,DR, 3,DR]</td>
<td>2.67</td>
</tr>
<tr>
<td>11</td>
<td>K</td>
<td>S10</td>
<td>3,DR,</td>
<td>3</td>
</tr>
<tr>
<td>13</td>
<td>R</td>
<td>S10</td>
<td>3, DR,</td>
<td>3</td>
</tr>
<tr>
<td>14</td>
<td>R</td>
<td>D14</td>
<td>3,DR,</td>
<td>3</td>
</tr>
<tr>
<td>15</td>
<td>K</td>
<td>D11</td>
<td>4,DR,</td>
<td>4</td>
</tr>
</tbody>
</table>

### Examiner Data for NEC Task 15: Activate Java Game

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>E</td>
<td>A12[D15]</td>
<td>1,PR,RS,TN,ER,FG,FM,[3,PR]</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>D51[S32]</td>
<td>2,DR,[3,DR]</td>
<td>2.5</td>
</tr>
<tr>
<td>3</td>
<td>T</td>
<td>[A10][D15]:D16</td>
<td>[3,TN,1N,DR,FM,[2,DR,TN,US]];3EN,1N,</td>
<td>2.67</td>
</tr>
<tr>
<td>4</td>
<td>T</td>
<td>[D14][S17]:[A9 S18]</td>
<td>[2,DR,TN,US[3,FM,DR];[3,1N,DR, 3,1N]</td>
<td>2.75</td>
</tr>
<tr>
<td>5</td>
<td>B</td>
<td>A33 D52</td>
<td>3,PR, 3,PR,</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>B</td>
<td>A34</td>
<td>4,US,</td>
<td>4</td>
</tr>
</tbody>
</table>

### Examiner Data for NEC Task 16: Delete Contacts

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>E</td>
<td>A13</td>
<td>1,TN,FM,ER,PR,FG,RS,</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>R</td>
<td>D16</td>
<td>1,DR,</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>R</td>
<td>S15</td>
<td>1,RS,?</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>T</td>
<td>D17[S19]</td>
<td>3,DR,FG[2,ER,CT,]</td>
<td>2.5</td>
</tr>
<tr>
<td>5</td>
<td>B</td>
<td>D53</td>
<td>3,DR,</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>M</td>
<td>S23</td>
<td>3,DR,</td>
<td>3</td>
</tr>
</tbody>
</table>

Page 480
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>K</td>
<td>A9 D13</td>
<td>2,RS,TN, 2,DR,TN,</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>K</td>
<td>D14</td>
<td>2,DR, LN,</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>R</td>
<td>D17 S16</td>
<td>3,DR, 3?</td>
<td>3</td>
</tr>
</tbody>
</table>

Examiner Data for NEC Task 17: Delete Photo

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B</td>
<td>A35 D54</td>
<td>2,DR, PR, FM, 2,DR, PR,</td>
<td>2</td>
</tr>
</tbody>
</table>

Examiner Data for NEC Task 18: Delete SMS Message

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B</td>
<td>D55[A36[S33]]</td>
<td>2,US,DR[3,US,CT,EF,PR[3,RS,]}</td>
<td>2.67</td>
</tr>
<tr>
<td>2</td>
<td>K</td>
<td>D15</td>
<td>3,US,</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>T</td>
<td>D18</td>
<td>3,VA,</td>
<td>3</td>
</tr>
</tbody>
</table>

Examiner Data for NEC Task 19: Turn Phone Off
### Samsung Tables

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>R</td>
<td>A1, D1</td>
<td>[S1] 3,FM,DR, 3,DR,PR, [2,FM,DR]</td>
<td>2.67</td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>S1</td>
<td>3,RS,</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>T</td>
<td>D1[S1]</td>
<td>3,PR[3,RS,]</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>B</td>
<td>A1[D1]</td>
<td>3,PR,RS,VA[4,PR,VA]</td>
<td>3.5</td>
</tr>
</tbody>
</table>

Examiner Data for Samsung Task 1: Turn the Phone On

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B</td>
<td>S2</td>
<td>2,DR,</td>
<td>2</td>
</tr>
</tbody>
</table>

Examiner Data for Samsung Task 2: Make a Call

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>K</td>
<td>D1[A1]</td>
<td>S1 1,US,PR,VA,FL,DR,LN,CS,[2,PR,CT,EF][2,DR]</td>
<td>1.67</td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>D3</td>
<td>2,CS,FL,VA,</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>R</td>
<td>D2</td>
<td>2,FM,VA,DR,</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>R</td>
<td>S2</td>
<td>2,DR,</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>T</td>
<td>D3[S3]</td>
<td>2,DR,US[3,CT,]</td>
<td>2.5</td>
</tr>
<tr>
<td>6</td>
<td>T</td>
<td>A1 S4</td>
<td>2,DR,PR,FM,LN, 3,DR,</td>
<td>2.5</td>
</tr>
<tr>
<td>7</td>
<td>E</td>
<td>A1[[S1][D2]]</td>
<td>2,LN,FM,DR[[3,DR,FM,][3,US]]</td>
<td>2.67</td>
</tr>
<tr>
<td>8</td>
<td>E</td>
<td>D1</td>
<td>3,DR,FM,</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>B</td>
<td>A3</td>
<td>3,RS,</td>
<td>3</td>
</tr>
<tr>
<td>10</td>
<td>M</td>
<td>D1 A1[S1]</td>
<td>4,DR,PR, 3,FM,PR,LN[2,DR,?]</td>
<td>3</td>
</tr>
<tr>
<td>11</td>
<td>M</td>
<td>D2 A2</td>
<td>3,FM,TN, 3,FL,FM,DR,</td>
<td>3</td>
</tr>
<tr>
<td>12</td>
<td>R</td>
<td>S3</td>
<td>3,DR,</td>
<td>3</td>
</tr>
<tr>
<td>13</td>
<td>R</td>
<td>S4</td>
<td>3,RS,</td>
<td>3</td>
</tr>
<tr>
<td>14</td>
<td>R</td>
<td>A2 D3</td>
<td>3,FM,DR, 3,DR, 3,RS,</td>
<td>3</td>
</tr>
<tr>
<td>15</td>
<td>T</td>
<td>D2 S2</td>
<td>3,DR,PR, 3,DR,</td>
<td>3</td>
</tr>
<tr>
<td>16</td>
<td>B</td>
<td>A2</td>
<td>4,LN,</td>
<td>4</td>
</tr>
<tr>
<td>17</td>
<td>R</td>
<td>S5</td>
<td>4,DR,</td>
<td>4</td>
</tr>
<tr>
<td>18</td>
<td>R</td>
<td>S6</td>
<td>4,DR,</td>
<td>4</td>
</tr>
</tbody>
</table>

Examiner Data for Samsung Task 3: Add Contacts to Phone Book

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>R</td>
<td>D4</td>
<td>3,DR,PR,</td>
<td>3</td>
</tr>
</tbody>
</table>

Examiner Data for Samsung Task 4 Make Call from Call History
### Examiner Data for Samsung Task 5: Access Web Browser

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>R</td>
<td>D6 S9</td>
<td>1,DR,?, 1,DR,</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>T</td>
<td>D4</td>
<td>1, LN, TN, CF</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>B</td>
<td>D4[S3]</td>
<td>2, FM, CT, [2, DR]</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>R</td>
<td>A3[D5[D8]]</td>
<td>3, RS, CT, [2, DR, VA[, D8]]</td>
<td>2.5</td>
</tr>
<tr>
<td>5</td>
<td>K</td>
<td>D2</td>
<td>3, DR, VA,</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>E</td>
<td>D4</td>
<td>4, DR,</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>K</td>
<td>A2</td>
<td>4, TN,</td>
<td>4</td>
</tr>
</tbody>
</table>

### Examiner Data for Samsung Task 6: Use Speed Dial

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>E</td>
<td>D6</td>
<td>1, US, LN,</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>D6</td>
<td>1, US, LN,</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>K</td>
<td>D3[A3[S2]]</td>
<td>1, US, DR, LN, CF[1, LN, DR[1, DR]]</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>R</td>
<td>D7[A4[S10]]</td>
<td>1, DR, PR, VA, CS, FM, [2, FM, DR, FL[2, DR, FM,]]</td>
<td>1.67</td>
</tr>
<tr>
<td>5</td>
<td>E</td>
<td>A3</td>
<td>2, DR, FM, RS, LN,</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>E</td>
<td>D7</td>
<td>2, CF</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>M</td>
<td>D3[A3]</td>
<td>2, US, DR, LN, CF, [2, PR, RS,]</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>E</td>
<td>D5[S3]</td>
<td>1,5, LN, PR, TN, FM, CF, VA[3, DR]</td>
<td>2.25</td>
</tr>
</tbody>
</table>

### Examiner Data for Samsung Task 7: Create and Save SMS

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>R</td>
<td>D10</td>
<td>1?</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>B</td>
<td>D7[S5]</td>
<td>2, LN, DR, PR[2, CS,]</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>E</td>
<td>A4 D8 S4</td>
<td>3, FM, DR, LN, 4, DR, 3, DR,</td>
<td>3.33</td>
</tr>
</tbody>
</table>

Page 483
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B</td>
<td>A5 D10[S8]</td>
<td>3, LN, RS, DR, FM, 1, US, DR, FM, PR[1,5,DR]</td>
<td>1.83</td>
</tr>
<tr>
<td>2</td>
<td>M</td>
<td>D4</td>
<td>2, DR, VA, LN, FM, 2, DR, US, 2, ER, CT,</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>R</td>
<td>D11 S13</td>
<td>3, DE, 2, DR,</td>
<td>2.5</td>
</tr>
<tr>
<td>4</td>
<td>E</td>
<td>D9 S5</td>
<td>3, DR, 3, DR, PR, 2, DR,</td>
<td>2.67</td>
</tr>
<tr>
<td>5</td>
<td>T</td>
<td>A2[D5][S6]</td>
<td>3, LN, DR, [3, DR, TN,][2, DR, RS,]</td>
<td>2.67</td>
</tr>
<tr>
<td>6</td>
<td>B</td>
<td>D9</td>
<td>3, DR,</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>M</td>
<td>A4 D5</td>
<td>2, PR, FM, 4, US, EF,</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>R</td>
<td>S14</td>
<td>3, RS,</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>T</td>
<td>S5</td>
<td>3?</td>
<td>3</td>
</tr>
<tr>
<td>10</td>
<td>K</td>
<td>A4[S3]</td>
<td>4, PR, RS, [4, RS, PR]</td>
<td>4</td>
</tr>
</tbody>
</table>

Examiner Data for Samsung Task 8: Use Calculator

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>K</td>
<td>D4[A5][S4]</td>
<td>1, US, FG, DR[1, EF, FG, RS,[2, RS,]]</td>
<td>1.33</td>
</tr>
<tr>
<td>3</td>
<td>R</td>
<td>D12[A6[S15]]</td>
<td>1, US, FG, RS,?[3, DR, FG[1?]]</td>
<td>1.67</td>
</tr>
<tr>
<td>4</td>
<td>B</td>
<td>S9</td>
<td>2, DR,</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>E</td>
<td>A5 D11</td>
<td>2, FG, PR, 3, DR,</td>
<td>2.5</td>
</tr>
<tr>
<td>6</td>
<td>B</td>
<td>D12</td>
<td>3, US,</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>M</td>
<td>D6</td>
<td>3, LN, DR,</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>M</td>
<td>A5[D7][D8]</td>
<td>3, EF, VA, FM,[[2, FG, LN, EF][4, DR]]</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>T</td>
<td>D7</td>
<td>3, US, FG,</td>
<td>3</td>
</tr>
<tr>
<td>10</td>
<td>T</td>
<td>A3[D8]</td>
<td>3, DR[3, TN, DR]</td>
<td>3</td>
</tr>
<tr>
<td>11</td>
<td>T</td>
<td>D6 S7</td>
<td>4, DR, 3, LN,</td>
<td>3.5</td>
</tr>
<tr>
<td>12</td>
<td>E</td>
<td>D10</td>
<td>4, LN, CS,</td>
<td>4</td>
</tr>
</tbody>
</table>

Examiner Data for Samsung Task 9: Assign Voice Recording

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>M</td>
<td>D9 A6</td>
<td>2, FG, EF, SM, 2, FG, ER, DR, PR,</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>E</td>
<td>S6[D12]</td>
<td>3, DR[2, DR, SM, FM, VA]</td>
<td>2.5</td>
</tr>
<tr>
<td>3</td>
<td>E</td>
<td>A6[D13 S7]</td>
<td>4, PR, DR, LN[2, US, 2, DR],</td>
<td>2.67</td>
</tr>
<tr>
<td>4</td>
<td>E</td>
<td>S8</td>
<td>3, DR,</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>B</td>
<td>A7 D13</td>
<td>2, US, PR, 4, US,</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>B</td>
<td>S10</td>
<td>3, DR,</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>M</td>
<td>D10 A7</td>
<td>4, US, FG, 2, FG, US, CT,</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>T</td>
<td>S8</td>
<td>3, RS,</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>T</td>
<td>A4[D9 S9]</td>
<td>3, FM, DR, RS, [3, DR, 3, RS,]</td>
<td>3</td>
</tr>
<tr>
<td>10</td>
<td>B</td>
<td>D14</td>
<td>4, US,</td>
<td>4</td>
</tr>
</tbody>
</table>

Examiner Data for Samsung Task 10: Create Appointment
### Examiner Data for Samsung Task 11: Use Voice Dial

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>T</td>
<td>D10</td>
<td>1,DR,US,EF,</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>D15 S11</td>
<td>3,DR, 2,RS,</td>
<td>2.5</td>
</tr>
<tr>
<td>3</td>
<td>M</td>
<td>A8 D11</td>
<td>3,DR,LN, 3,US,LN,</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>B</td>
<td>A8 D16</td>
<td>4,DR,US, 4,US,</td>
<td>4</td>
</tr>
</tbody>
</table>

### Examiner Data for Samsung Task 12: Delete Appointment

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>M</td>
<td>A9 D12</td>
<td>2,US,CT,FG, 3,US,DR,</td>
<td>2.5</td>
</tr>
<tr>
<td>2</td>
<td>M</td>
<td>D13</td>
<td>3,DR,PR,</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>R</td>
<td>S16</td>
<td>3,DR,RS,?</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>E</td>
<td>D14</td>
<td>4,LN,CT,</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>B</td>
<td>D17</td>
<td>4,US,</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>R</td>
<td>D13</td>
<td>4,DR,</td>
<td>4</td>
</tr>
</tbody>
</table>

### Examiner Data Samsung Task 13: Send Saved SMS

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B</td>
<td>D20</td>
<td>1,US,PR,DR,</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>M</td>
<td>A12 D16</td>
<td>1,DR,EF,SM, 1,US,LN,</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>K</td>
<td>A6 D5[S5]</td>
<td>2,FM,PR,CS, 1,US,LN,PR,TN,[2,LN,CS,]</td>
<td>1.67</td>
</tr>
<tr>
<td>4</td>
<td>E</td>
<td>A7[D15]</td>
<td>3,FM,PR[1,DR,PR,US]</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>M</td>
<td>A11 D15</td>
<td>3,PR,FM, 2,DR,PR,TN,LN,</td>
<td>2.5</td>
</tr>
<tr>
<td>7</td>
<td>E</td>
<td>S9</td>
<td>3,DR,</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>M</td>
<td>A10[D14]</td>
<td>3,DR,PR[3,PR,DR]</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>T</td>
<td>A5</td>
<td>3,DR,FM,</td>
<td>3</td>
</tr>
</tbody>
</table>

### Examiner Data for Samsung Task 14: Delete Contact

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>M</td>
<td>D17</td>
<td>2.5, ,US,</td>
<td>2.5</td>
</tr>
<tr>
<td>2</td>
<td>R</td>
<td>S18</td>
<td>3,RS,</td>
<td>3</td>
</tr>
</tbody>
</table>

Page 485
Appendix F

APPENDIX F: Mobile Phone NGOMSL Model

NGOMSL Step B – The Recursive Procedure

This second step of the NGOMSL procedure is a recursive procedure for building methods, checking them against assumptions and moving to new levels of detail if necessary. This is a cycle that must be repeated several times, in order to refine a crude NGOMSL model to one that is complete, natural and consistent. These iterations are called consistent with Kieras’s (1997) documentation on the NGOMSL method. The remainder of this appendix documents the development of the NGOMSL model across several of these passes, which due to the extensive nature of this model are not all discussed.

Step B1 – Drafting Methods

The first step of the recursive method is to start building methods that the user will follow in order to achieve each of the goals specified in the previous step. Based on the selection rule drafted in the last section, there are seven goals for which methods (and or selection rules) must be drafted. These are make call, use text message, use an accessory, review contacts, access a web application and use media. Thus seven methods (or selection rules), must be drafted, one for each of these.

*Kieras provides several recommendations for drafting these methods:*

1. No specific keystroke sequences for anything higher than four levels down
2. Move to a new level of detail if the number of steps exceeds five.

Thus no more than five steps, and no primitive operators (e.g. keystrokes) are used in any of the methods at this level of detail. Kieras provides four additional rules based on earlier work with Bovair, Kieras and Polson (1990). These rules are also used adhered to by this study and are as follows:

1. No more than one “accomplish goal” operator to a step
2. No more than one “high level operator to a step
3. For ordinary or novice users use no more than one external primitive operator to a step
4. For expert users, it is possible to use multiple external operators in a step, where no sub-goals or decisions exist.
This means that only one method or selection rule, or a high level operator (not a sub-goal) can only be called on a single NGOMSI step. Since this is a cognitive based technique, Kieras acknowledges a difference in behaviour between expert and novice users. These last two rules will be addressed later in this section. The first of the goals addressed in this GSE step is activate phone, which is an essential prerequisite for every mobile phone available.

Method for Goal: Activate Phone
Assumptions: phone can be opened and closed and user is aware phone can be opened and closed
1. Decide if phone is closed then open phone
2. Decide if phone is off then accomplish goal: turn on phone
3. Decide if password on then accomplish goal: set password
Return with goal accomplished.

There are two assumptions as listed in the method above. First of all the phone can be opened and secondly that the user is aware it can be opened. This model is intended for a novice user who is either initially aware it can be opened, discovers this in the manual, or obtains this through a short amount of trial and error. Either way, the operators involved in physically opening the device, is not of great importance to this study. This is on the basis that once a user has opened it, they will not soon forget this. The two assumptions in the activate phone method are sufficient.

Making Calls
The first method is the perform phone task is make call. Since a decision between methods must also be made here, an additional selection rule, and not a method, is drafted. This is consistent with Kieras (1997) who states that if “the analyst discovers that there is more than one method to accomplish a goal then the general goal should be decomposed into a set of specific goals one for each method.” (Kieras 1997: 741)

Selection Rule for Goal: Make Call
Assumption: These methods activate a standard mobile phone call. This excludes, video calls, which are now available on more recent model phones such as the NEC used in the study.
If call is the last to be received and number or contact is displayed on phone screen, then accomplish goal Return last call
If user knows number, and is not aware of short cut then accomplish goal Make Manual Call
If user cannot recall number, but has assigned name to contacts, then accomplish goal Make Call from Contacts
If user has assigned short cut key to phone number, and can recall it then accomplished goal Use Short Cut (e.g. Speed Dial).
If user has specified a voice activated call for the target, then accomplish goal Make Voice activated call.
If user has received or made a call previously, but cannot recall number and number is not in contacts, then accomplish goal Make a Call from Call History/Register.
If user cannot recall number, and knows that SMS message from target has been received, and that number is not a secret number then accomplish goal Make a Call from SMS message

Return with goal accomplished

Several assumptions must now be specified. Firstly that all seven techniques are available to the user from the outset and secondly that the user is able to use all them based on the conditions specified as above. The Methods drafted for these seven goals are as follows:

**Make Call Sub Methods in NGOMSL (Pass1)**

**Method for Goal: Return last call**
1. AG: Get Last Number
2. AG: Activate call
3. Return with goal accomplished

**Method for Goal: Call from contacts**
Assumptions: Phone records missed, dialled and received calls. User has received, dialled or missed a call related to this phone number.
1. Accomplish Goal Open Contacts / Phone Book
2. Accomplish Goal: Access Contacts
3. Accomplish Find Name in Contacts
4. Accomplish Goal Activate Call
5. Return with goal accomplished

**Method for Goal: Make a Manual Call**
1. Accomplish Goal: Get number
2. Accomplish Goal: Enter Number
3. Activate Phone Call
4. Return with goal accomplished

**Method for Goal: Make a Call from Call History**
1. Recall Name and previous call
2. Access Call register / history
3. Decide if call is in accomplish goal: get number from calls dialled
4. Decide if call is in accomplish goal: get number from calls received
5. Decide if call is in accomplish goal: get number from calls missed
6. Activate Call

**Method for Goal: Use Speed Dial**
1. Recall name
2. Recall button combination
3. Press Buttons
4. Verify Call started
5. Return with goal accomplished

**Method for Goal: Make a Voice Activated Call**
Assumptions: There is some kind of mode the phone must be in to make voice calls. Secondly when the name is spoken the call is instantaneous (call button need not be pressed). The user may repeat name (on mistake) without having to activate voice mode.
1. User recall name
2. User Activate Voice Mode
3. Say Name
4. Verify if number activated then return with goal accomplished
5. Go to 3

Looking at the methods above, it becomes clear that many of the methods share a common pattern. First the target is chosen, the number is obtained, it is entered into the phone and the call is activated.
PASS 2

On a second pass of the Make Call methods significant changes are made to make the technique more natural. The make call method is now broken into a simple four step method, including a new high level operator “converse” which is not defined here, owing to its complexity. The start call method now distinguishes between a standard call, where the user has to find a number, and the two short cut methods, make a voice activated call, and make a speed dial call, where the number is allocated to a automatic activating preset. The alternative of course is to start a standard phone call. Where a user gets a number (wither entering it themselves or finding it in the phone) , and then activates the call.

<table>
<thead>
<tr>
<th>Make Call Methods: PASS 2</th>
<th>Selection Rule for Start Call</th>
</tr>
</thead>
<tbody>
<tr>
<td>Make a Phone Call</td>
<td>If number is stored as Voice then</td>
</tr>
<tr>
<td>Start Call</td>
<td>make Voice activated Call</td>
</tr>
<tr>
<td>Converse</td>
<td>If number is stored as Speed Dial then</td>
</tr>
<tr>
<td>End Call</td>
<td>make Speed Dial Call</td>
</tr>
<tr>
<td>Return with goal Accomplished</td>
<td>If number is not stored as shortcut then</td>
</tr>
<tr>
<td></td>
<td>Start a standard phone Call</td>
</tr>
<tr>
<td></td>
<td>Return with goal accomplished</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Selection Rule for Choose Number</th>
<th>Return with goal accomplished</th>
</tr>
</thead>
<tbody>
<tr>
<td>If number must be entered manually then</td>
<td></td>
</tr>
<tr>
<td>input number manually</td>
<td></td>
</tr>
<tr>
<td>If number is in SMS then</td>
<td></td>
</tr>
<tr>
<td>input number from SMS</td>
<td></td>
</tr>
<tr>
<td>If number is in Contacts then</td>
<td></td>
</tr>
<tr>
<td>input number from contacts</td>
<td></td>
</tr>
<tr>
<td>If number is in Call History then</td>
<td></td>
</tr>
<tr>
<td>get number from recorded calls</td>
<td></td>
</tr>
</tbody>
</table>

PASS 3

The implications of the above changes are that new methods are now needed to describe actually getting a phone number and entering it into the phone. Many of these methods share similarities (they each involve the user making their way to a screen, and then searching the entries for a particular number). These new methods are as follows:

<table>
<thead>
<tr>
<th>Make Call Methods PASS 3</th>
<th>Input number from Contacts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input number from SMS</td>
<td>Input number from Contacts</td>
</tr>
<tr>
<td>1. Go to Messages</td>
<td>1. Go to Contacts</td>
</tr>
<tr>
<td>2. Go to Inbox</td>
<td>2. Search Pages for Name</td>
</tr>
<tr>
<td>3. Search Pages for Name</td>
<td>3. Get Number</td>
</tr>
<tr>
<td>4. Get Number</td>
<td>4. Return with goal accomplished</td>
</tr>
<tr>
<td>5. Return with goal accomplished</td>
<td></td>
</tr>
</tbody>
</table>
Input number from Recorded Calls
1. Go to Call Records
2. Decide if a call from this number was missed then Go to Missed Calls
3. Decide if a call from this number was dialled then Go to Dialed Calls
4. Decide if call from number was answered Go to Received Calls
5. Search Entries for number
6. Activate Call
7. Return with goal accomplished

Make voice activated Call
1. Start Voice Recording
2. Say Name
3. Verify number ringing
4. Return with goal accomplished

Make a speed dial call
1. Choose Recipient
2. Retrieve shortcut key from LTM
3. Hold down key
4. Verify number ringing
5. Return with goal accomplished

Entering a number manually is significantly different. The phone number is not stored within the phone, and must be obtained through either recall from memory or actually reading the number from another source. A Decide operator is used here, to decide whether the number is obtained by reading digit by digit or by remembering the number, which requires access to Long Term Memory. These digits must be stored in working memory such that they may be held long enough to be keyed into the phone. Recall and Retrieve are mental operators that reply to Long Term Memory (LTM) and Working Memory (Respectively).

Method for goal: Input Number Manually
Assumption user reads phone number as a whole. However when reading, each number is read separately.
1. Decide if number is in memory then Recall Number and retrieve from LTM the next digit and retain
2. Decide if number is read then read next digit
3. Retain digit
4. Accomplish goal Enter next digit in number
5. Forget digit
6. Recall next digit
7. Decide if any more digits go to 2
8. Return with goal accomplished
PASS 4

Method for Goal Search Entries for Number
Assumptions user moves through records 1, by 1 looking for entry
1. Recall name for recipient from LTM and retain
2. If name matches current record forget name and return with goal accomplished
3. Decide if name before – Move to previous record
4. Decide if name after – Move to next record
5. Go to 2

Assuming that this behaviour is based on an actual device, there must be methods for navigating throughout the phone. An assumption is made here, that the phone employs a central menu system and the standard phone key layout. Since the phone functions must be supported by an underlying architecture. It is important to specify a number of more detailed methods on which the phone interface will be based.

Other Low Level Functional Goals and Operations
These methods that follow are based on the assumption that the phone is divided in functionality across screens and a central menu system. It is also assumed that the mobile has a visual means of picking items on screen and choosing between menu options. Clearly no mouse is available so navigation between items involves directional keys and some visible change in the display when an item is picked.

Go to Destination (Screen)
1. Identify Destination
2. Decide if know short cut then AG: use short cut (key presses)
3. Else Navigate Menu to destination
4. Return with goal accomplished

Method for goal: Navigate Menu to Destination
1. Recall command name and retrieve from LTM the name of next top level menu
2. Decide if command in current menu then Select command from menu
3. Else if command in sub menu then retrieve from LTM the next sub menu name
4. Retain the next menu name, and Select <next menu> from menu
5. Return with goal accomplished

Method Pick Command from Menu
1. Decide if command not highlighted then AG: Move marker to command
2. Locate Select Button
3. Move hand to select Button
4. Press Select Button
5. Return with goal accomplished

Method for goal: Key Press
1. Recall Key to Press
2. Locate Key
3. If finger not on key Move finger to Key
4. Press Key
5. Return with goal accomplished

Method for goal: Pick Digit
1. Press key
2. Verify character on screen
3. Decide if key does not match target then go to 1
4. Return with goal accomplished
between the keypad keys
1. Decide if this digit same as last key press
   then wait for cursor
2. Recall next digit from WM
3. Decide if Digit is different locate
   corresponding key on keypad
4. Decide if hand not on key move finger to key
5. Pick Digit.
6. Verify that digit has been pressed
7. Return with goal accomplished

Method for Scroll to Destination
1. User move hand to Up Key
2. Press UP key
3. Verify that highlight has moved up
4. Return with goal accomplished

Method Move pointer to Icon
1. Locate Icon
2. Move highlight towards icon
3. If highlight is on icon then return with
   goal accomplished
4. Go to 2

Several of these methods breach the 5 steps and less restriction suggested by Kieras. Since this is at the
lowest level of detail, that is the author does not want to expand these any further, it is possible to
exceed the five steps by a small margin.

Text Message functions
With the make call methods established the recursive procedure must also be applied to the users SMS
text goals. In this first pass, a high level goal “use text message” is used to specify a selection rule for
the users tasks. Methods are also needed for sending reading and making SMS text messages.

Text Message related NGOMSL Methods (Pass 2)

Selection Rule for goal Use Text Message
If the task is making a new SMS message
If the task is sending a Text Message
If the task is reading a Task Message
Return with goal accomplished

Method for goal Read a Text Message
1. Access SMS message
2. Find Message
3. Open Message
4. Read Message
5. Close Message
6. Return with goal accomplished

Method for goal Send a Text Message
1. Method for goal Send a Text Message
2. Access SMS messages
3. Open a Message
4. Choose Number
5. Send Message
6. Verify that Message is Sent
7. 5. Return with goal accomplished

Method for goal Make Text Message
1. Access SMS messages
2. Open New SMS message
3. Enter Message
4. Save Message
5. Return with goal accomplished
It is possible to construct a goal that better represents the “use text” function. First of all, the user must go to wherever the SMS messages functions are stored within an interface. They must then open a message, be that an existing message or a new message.

There are several assumptions that need to be specified on the differences between received and composed messages. Incoming messages cannot be edited. There is no point in editing a received message. It may however be forwarded to a new recipient, or replied to which creates a new message in reply, or saved as an editable messages. On the other hand Messages composed by the user are editable. They may be saved, and edited. Edited is defined as altering the content of a message.

Use Message
Assumptions – Messages can be new, open or saved
1 Go to Screen Messages
2 Decide if incoming message then Open Message [Received]
3 Decide if existing message then Open Saved Message
4 Decide if new message then Make New Message
5 Perform Message Task
6 Return with goal accomplished

The method above indicates that there are three possible goals when using a message. The first is to open a received message. A second possibility is to open a saved message a message that the user has composed) and saved, like a draft. The third option is to create a new message. The use message goal above exceeds five steps. In the next pass, it is noted that steps 2 to 3, all involve the user opening or acquiring a type of message. They may all be bound into a common sub-goal reducing the steps at the above level. The revised model is now:

Use Message
Assumptions – Messages can be new, open or saved
1 1 Go to Messages folder
2 2 Open Message
3 Perform Message Task
4 Decide if no more tasks then return with goal accomplished
5 6 Return with goal accomplished

Selection Rule for Open Message
Decide if user wants a new message then accomplish goal
open a received message
Decide if user wants an existing message then accomplish goal
open a saved message
Decide if user wants an new message then open a New Message
Return with goal accomplished
Moving to the next level of detail, methods for opening a received and saved message are as follows:

**Open a Received Message**
- Go to Received Messages Folder
- Find Message
- Return with goal accomplished

**Open a Saved Message**
- Go to Drafted Messages Folder
- Find Message
- Return with goal accomplished

Using a form consistent with high-level goals, performing a message related task is assigned to a selection rule.

**Selection Rule for Perform Message Task**
1. If the task is open message then open message then AG: open message
2. If the document is editable and the task is read message then AG read message
3. If the document is editable and the task is edit message then AG: goal edit message
4. If the document is not editable and the task is to send message then AG: goal forward message
5. If the document is not editable and the task is to save message then AG: Save copy
6. If the task is to save message then AG: save message
7. If the task is send message then AG: send message
8. Return with goal accomplished

Open message is repeated here, because the user may choose to open another message. While this may not seem natural, it is meant to reflect the users’ decision in relation to a mobile phone and is kept here for completeness.

**SEND FUNCTIONS (PASS 4)**

<table>
<thead>
<tr>
<th>Selection Rule for Send Message</th>
<th>Read an Open Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Decide if message not open then AG: Open Message</td>
<td>1. Decide if message not open then open Message</td>
</tr>
<tr>
<td>2. AG: Choose Number</td>
<td>2. Read Text</td>
</tr>
<tr>
<td>3. Issue Command SEND</td>
<td>3. Decide if more text off screen then scroll Text</td>
</tr>
<tr>
<td>4. Verify message sent</td>
<td>4. Return with goal accomplished</td>
</tr>
<tr>
<td>5. Return with goal accomplished</td>
<td></td>
</tr>
</tbody>
</table>

**Selection Rule for Save Message**
1. Decide if message not open then AG: Open Message
2. Issue Command SAVE
3. Verify message saved
4. Return with goal accomplished
5. Return with goal accomplished

**Edit Message**
1. Decide Next Task
2. Decide if not at location then AG: move to Task Location
3. Perform Text Function
4. If no more tasks return with goal accomplished
5. Go to 1
EDIT RELATED FUNCTIONS (PASS 5)

SMS
Selection Rule for Perform Text Function
If the Task is Enter text then AG Enter Text
If the task is delete text then AG Delete Text
Return with goal accomplished

Move to Task Location
1. Recall Task Location
2. Scroll cursor position to destination with keys
3. Return with goal accomplished

Method for Edit Text
1. Decide if not at location move to text location
2. Edit Text
3. Decide if no more changes return with goal accomplished
4. Go to 2

Calculator

The next set of methods relate to the accessories (the reminders and calculators). These are defined here as tools that serve non-essential functions. The first of these goals is "the calculate an equation goal". This is a relatively simple task. The users need only open the calculator, enter an equation, get the answer and the goal is accomplished

Method for goal: calculate an equation
1. User Access calculator
2. Accomplish goal: Enter equation
3. Get Answer
4. 3. Return with goal accomplished

In pass 2, the enter equation and get answer methods are stated. These are as follows:

Method for goal: Enter an Equation
1. Enter first digit
2. Enter operator
3. Enter next digit
4. Verify equation
5. Decide if no more digits return with goal accomplished
6. go to 2

Method for goal: Get Answer
1. Press = Key
2. Verify Answer
3. Return with goal accomplished

In Pass 3, the enter equation method is altered to show WM usage and the enter operator method is specified.
Method for goal enter equation
1. Retrieve Equation from LTM and recall first number
2. Recall Operator and Enter Operator
3. Decide if only one number, or more numbers than two go to 1
4. Verify Equation
5. Return with goal accomplished

Method for goal Enter Operator
1. Retain Operator
2. Decide if operator is divide then Press Button EQUALS
3. Decide if operator is multiply then Press Button MULTIPLY
4. Decide if operator is addition then Press Button PLUS
5. Decide if operator is subtraction then Press Button MINUS
6. Return with goal accomplished

Reminders

Reminders are messages stored and allocated to a date, such that they pop up, when that date is reached. In the study, reminders are set and deleted. Therefore the top-level goal is set to Manage Reminders to keep it as high level as possible. PASS 1 is as follows

PASS 2
Method for Goal Manage Reminders
1. Go to Reminders
2. Go to Date
3. Perform Reminder Task
4. Decide if any more reminder tasks Return with goal accomplished
5. Go to 4

Selection Rule for Perform Reminder Task
If user wants to edit and existing Reminder accomplish goal edit a Reminder
If user wants to set a new reminder
Accomplish goal edit reminder
If user wants to enter a new reminder
Accomplish goal: add a reminder
Return with goal accomplished

Add Reminder
1. Create New Reminder
2. Edit Text
3. Save Reminder
4. Return with goal accomplished

Edit Reminder
1. Choose Reminder
2. Open Reminder
3. Edit Text
4. Save Reminder
5. Return with goal accomplished

Delete Reminder
1. Choose Reminder
2. Open Reminder
3. Cancel Reminder
4. Verify Deletion
5. Return with goal accomplished

PASS 3

Choose Reminder
1. Retrieve Message from LTM and retain
2. Search Entries for Message
3. Issue Command Open
4. Verify Reminder
5. Return with goal accomplished

Edit Reminder
1. Choose Reminder
2. Edit Text
3. Issue Command Save
4. Verify Reminder
5. Return with goal accomplished
Playing Games

The Play a game method is a detailed one. However, this study does not cover the dynamics of actually playing a specific game. In reality, this study covers, finding starting and ending a game. The actual game in the user study Play Golf, must also be downloaded from the web, but the actual download

Method for goal Start A Game
1. Go to Games
2. Accomplish goal: Find Game
3. Accomplish goal: Activate a game
4. Play Game
5. Quit Game
6. Return with goal accomplished

<table>
<thead>
<tr>
<th>Play a Game Methods [PASS 3]</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Method for goal Activate Game</strong></td>
</tr>
<tr>
<td>1. Issue Start Command</td>
</tr>
<tr>
<td>2. Verify Game started</td>
</tr>
<tr>
<td>3. Return with goal accomplished</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

Web Related Methods

This study included 3 tasks specifically. The first was to connect to the web. The second was to download a game, and the third task was to compose and send an email message.

Method for Goal: Browse Web
1. Activate Web Browser
2. Browse web
3. Return with goal accomplished

Method for Goal: Compose Email
1. Create new email
2. Enter email message
3. Choose address
4. Send Email

Method for Goal Download Game
1. Open Web Browser
2. Find Games Site
3. Choose Game
4. Start Download
5. Wait until download finished
6. Confirm download
7. Return with goal accomplished
8. Go to 2

In Pass 3 these three methods were refined by recognising that both the games download site and the email entry site are both web sites. The revised and final model for web activity in this study now is as follows:
Pass 3
Connect to Web
1. Go to Internet Connection
2. Activate Internet Connection
3. Verify connection
4. Return with goal accomplished

Download Application Game
1. Find Site Games
2. Access Games Site
3. Download Game
4. Verify Download
5. Return with goal accomplished

Access Web Based Email Site
1. Enter Address
2. Enter Message
3. Send Email

Use Web Services
1. Connect to Web
2. Decide if web browser not open open
   Web Browser
3. Access Web Site
4. Decide if any more web tasks then return
   with goal accomplished
5. Go to 3.

Multimedia Related Methods

The multimedia methods are specific to the NEC used in the study, since the Samsung model was not equipped with a camera (relate to the use of the Video Camera). After several passes, the final multimedia model is as follows. Many of these are externally and analyst defined operators since these are suitable for describing a natural sequence of operations.

Method for Take Photo
1. Access Camera
2. Line up Target
3. Take photo
4. Return with goal accomplished

Method for Take Video
1. Access Video Camera
2. Line up Target
3. Start recording
4. Follow Target
5. Decide finish recording stop recording and
6. Return with goal accomplished
7. Go to 4.

View Media
1. Access Medial Library
2. Find Media
3. Open and View Media

Delete Media File
1. Access Media Library
2. Find Media
3. Open Media
4. Delete Media

Contact Related Methods

Contact related methods are those associated with the mobile’s “phonebook” or “Contacts” list. The tasks in this study, involved adding deleting contacts and calling contacts. Using contacts to make a...
call is already covered in the make call methods earlier. The list below covers the methods extracted over several NGOMSL passes.

Method for goal Add Contact
1. AG: Access Contacts
2. AG: Create New Contact
3. AG: Save Contact
4. Return with goal accomplished

Method for goal Delete Contact
1. Access Contacts
2. Find Contact
3. Delete Contact
4. Return with goal accomplished

Method for Goal Find Contact
1. Recall Name
2. If contact does not match contact move to next?
3. If contact matches name return with goal accomplished
4. Go to 2

Create New Contact
Open New Contact
Enter Name
Enter Number
Return with goal accomplished

Edit Contact
1. Open Existing Contact
2. Edit Details
3. Save Contact
4. Return with goal accomplished

Enter Text Field
1. Recall DATA
2. Decide if at field else scroll to field
3. Type DATA in
4. Verify Text
5. Return with goal accomplished

Create New Contact
Open New Contact
Enter Name
Enter Phone number
3. Verify Details
Return with goal accomplished
## APPENDIX G: Surveys

### VCR Study: Questionnaire

<table>
<thead>
<tr>
<th>Name:</th>
<th>Age:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gender:</td>
<td>Occupation:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>1. Do you own a video recorder?</th>
<th>2. Do you have free access to a video recorder</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>3. The brand?</th>
<th>4. The model?</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>5. How often do you use your VCR? Please circle one.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rarely if ever</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>6. How many VCRs have you used?</th>
<th>7. How many VCRs have you owned?</th>
</tr>
</thead>
</table>

8. If you wanted to record a program while you were out how would you do it?

<table>
<thead>
<tr>
<th>9. Do you know what timer recording is?</th>
<th>10. Have you ever used timer recording?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>11. Do you know what G-CODE is?</th>
<th>12. Have you ever used G-Code recording?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>13. How difficult would you describe timer recording to be (Please circle one)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very Difficult</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>14. How often do you use timer record?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rarely if ever</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>15. How did you learn how to timer record. (Please Circle).</th>
</tr>
</thead>
<tbody>
<tr>
<td>From Manual</td>
</tr>
</tbody>
</table>

Other:
16. Which of the following things have you done on a VCR.

<table>
<thead>
<tr>
<th>Played a Tape</th>
<th>Recorded a program</th>
<th>Set the VCR clock</th>
<th>Tuned the VCR Channels</th>
<th>Set VCR GCode settings</th>
</tr>
</thead>
</table>

17. I accept that this information and the videotaped sessions (over the next three pages) will be used for PhD Study and related academic purposes.

Signature:
# Mobile Study: Questionnaire

<table>
<thead>
<tr>
<th>Name:</th>
<th>Age:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gender:</td>
<td>Occupation:</td>
</tr>
</tbody>
</table>

1. Do you own a Mobile Phone?  
   Yes | No

2. What is the Make (e.g. Erikson, Nokia...)?

3. The Model? E.g. 3310

4. The network (Optus, Vodafone...)

5. How long have you owned this phone?  
   | Less than a month | Less than Three Months | Less than Six months | Less than a year | More than a year | More than two years | More than three years |
|-------------------|---------------------|-----------------------|---------------------|------------------|-------------------|----------------------|-----------------------|

6. Is this your first phone?  
   Yes | No

7. If No: How many have you owned?

8. If no: For what period of time have you owned a mobile phone?  
   | Less than a month | Less than Three Months | Less than Six months | Less than a year | More than a year | More than two years | More than three years |
|-------------------|---------------------|-----------------------|---------------------|------------------|-------------------|----------------------|-----------------------|

9. If you have owned other phones, can you remember what they were? (Please List)

10. If you have owned a previous phone, why did you change to your current one?  
    | New Plan | Lost or stolen | Broken | Changed network | Disliked phone | Wanted new features e.g. web access |
|-------|-------------|---------|--------|-----------------|---------------|---------------------------------|

11. How often do you use your phone to make calls approximately? (blank if never)  
    | Rarely if ever | Once a year | Several times a year | Once a month | Several times a month | Once a Week | Several Times a week | Once a day | Several times a day | Many times in a day |
|-------|---------------|-------------|---------------------|-------------|----------------------|-------------|----------------------|------------|----------------------|---------------------|

12. How often do you use your phone to receive calls approximately? (blank if never)  
    | Rarely if ever | Once a year | Several times a year | Once a month | Several times a month | Once a Week | Several Times a week | Once a day | Several times a day | Many times in a day |
|-------|---------------|-------------|---------------------|-------------|----------------------|-------------|----------------------|------------|----------------------|---------------------|
12. How often do you send SMS messages approximately? (leave blank if never)

<table>
<thead>
<tr>
<th>Rarely if ever</th>
<th>Once a year</th>
<th>Several times a year</th>
<th>Once a month</th>
<th>Several times a month</th>
<th>Once a Week</th>
<th>Several Times a week</th>
<th>Once a day</th>
<th>Several times a day</th>
<th>Many times in a day</th>
</tr>
</thead>
</table>

13. How often do you receive and read SMS messages approximately? (blank if never)

<table>
<thead>
<tr>
<th>Rarely if ever</th>
<th>Once a year</th>
<th>Several times a year</th>
<th>Once a month</th>
<th>Several times a month</th>
<th>Once a Week</th>
<th>Several Times a week</th>
<th>Once a day</th>
<th>Several times a day</th>
<th>Many times in a day</th>
</tr>
</thead>
</table>

How do you usually make a call? If you use more than one, please indicate preference by marking boxes 1 (highest) through to 5 (lowest). You may leave unused options unranked.

<table>
<thead>
<tr>
<th>Keyed in numbers</th>
<th>Accessed numbers in phone storage</th>
<th>Used Speed Dials</th>
<th>Used number listed in call history</th>
<th>From SMS number</th>
<th>Used/voice activated calls</th>
</tr>
</thead>
</table>

If other please indicate:

15. How did you learn how to operate your current phone? (Please Circle).

- From Manual
- Figured it out
- Someone showed me
- It was very similar to a previous phone

Other:

16. Have you ever owned or used a Samsung SGH-T100

- Yes
- No

17. Have you ever owned or used a NEC e606?

- Yes
- No
17: Which of the following mobile phone functions have you used?

<table>
<thead>
<tr>
<th>Added modified or Erased number in phone book</th>
<th>Set the clock</th>
<th>Used the Alarm Clock</th>
<th>Set reminders</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accessed voice mail</td>
<td>Accessed Web Browser</td>
<td>Downloaded content from web</td>
<td>Linked phone to computer to transfer files</td>
</tr>
<tr>
<td>Edited or created a picture</td>
<td>Composed a ring tone</td>
<td>Played a Game</td>
<td>Used the calculator</td>
</tr>
<tr>
<td>Set a password or pin number</td>
<td>Set the key-lock</td>
<td>Set ring tone</td>
<td>Viewed missed calls</td>
</tr>
<tr>
<td>Sent a picture message</td>
<td>Used camera for photos or videos</td>
<td>Accessed missed/received calls</td>
<td>Set phone volume</td>
</tr>
<tr>
<td>Barred Calls</td>
<td>Diverted Calls</td>
<td>Made a multi-party call</td>
<td>Set language</td>
</tr>
<tr>
<td>Sent or saved multimedia files (sound, pictures, video)</td>
<td>Drawn picture</td>
<td>Set background picture</td>
<td>Used Voice Activated Calls</td>
</tr>
</tbody>
</table>

18. How easy would you say your phone is to use?

<table>
<thead>
<tr>
<th>Very Difficult</th>
<th>Difficult</th>
<th>Average</th>
<th>Easy</th>
<th>Very Easy</th>
</tr>
</thead>
</table>

Page 505
19. I accept that this information and the videotaped sessions (over the next three pages) will be used for PhD Study and related academic purposes. I also accept that these tapes will be retained for analysis and examination. I also accept images from the video footage will also be used or published as part of the PhD Thesis for which this research is being conducted.

Signature:
APPENDIX H:
Mobile Phone NGOMSL model some conversion examples

Converting the NGSOML phone model to GSE

The NGOMSL phone model in this chapter is a great deal more detailed, and extensive than the VCR model built in the previous section. Whereas Kieras’s VCR model contains methods and selection rules for only 7 Goals, the Mobile phone model must address 64. This also makes the GSE modelling in this section, a much larger task. The aim of this section is to demonstrate NGOMSL conversion, at differing levels of details. The models in this section are based on different passes of the NGOMSL method as shown in APPENDIX D.

GSE models for Top Level Use Phone Goals

The first two goals follow the template specified by Kieras’ for the top-level goals. This means a method for user phone and a selection rule for perform phone task

<table>
<thead>
<tr>
<th>NGOMSL Representation</th>
<th>GSE Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method for goal: Use Phone</td>
<td>\begin{itemize} \item \text{DECIDE: if phone is off, achieve goal: \texttt{activate phone}} \item \text{DECIDE: if call is incoming \textbf{Accomplish goal: \texttt{respond to call}}} \item \text{DECIDE: if no more tasks then return with goal accomplished \textbf{Accomplish goal: \texttt{Perform Phone Task}}} \end{itemize}</td>
</tr>
</tbody>
</table>

NGOMSL Method for Goal: Use Phone

Page 507
The final of the top-level goals is the activate phone method called from the Use phone method. Initially most of what happens here is described by user-initiated events, since at this stage of the process, the model is the user's high-level intentions.

**GSE models for Making Phone Calls**

There are at least five different ways of making phone calls between the two phone models and at least as many methods.
Methods for General functionality
### Pick Command From Menu

1. **GOAL**: Touch to Display Screen??

2. **USER**: Which Command Not Highlighted??

3. **USER**: Yes??

4. **USER**: Which Select Button??

5. **USER**: Please Select Button??

6. **USER**: There needs to Select Button??

7. **USER**: Press SELECT Button??

8. **USER**: Notify Service Change??

9. **GOAL**: Accomplished

### Key Press

1. **GOAL**: Key Press??

2. **USER**: "Send NXY You Press??

3. **USER**: "End Code??

4. **USER**: "Decide If Right and on Key??

5. **USER**: "Decision if Finger on Key??

6. **USER**: "Finger to Key??

7. **USER**: "Wipe Key??

8. **USER**: "Workflow Key??

9. **GOAL**: Accomplished

### Enter Character

1. **GOAL**: Outside Character??

2. **USER**: "Check digit times as bit key pressed??

3. **USER**: "Here??

4. **USER**: "What is Current to move??

5. **USER**: "Move key??

6. **USER**: "Push Key??

7. **USER**: "Move Finger to Key??

8. **USER**: "Press Finger to Key??

9. **USER**: "Push Light??

10. **GOAL**: Accomplished
Text Message Methods
Web Related Methods
Multimedia Related Methods
Contact Related Methods
<table>
<thead>
<tr>
<th>Create New Contact</th>
<th>Edit Contact</th>
<th>Enter Details</th>
<th>Enter Text Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>GOAL: Create New Contact?</td>
<td>USER: First Name?</td>
<td>GOAL: Accomplished</td>
</tr>
<tr>
<td>2</td>
<td>GOAL: Open Existing Contact?</td>
<td>USER: Contact Name?</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>USER: Write Details?</td>
<td>USER: Phone Number?</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>GOAL: Save Contact?</td>
<td>USER: Email Details?</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>GOAL: Accomplished</td>
<td>GOAL: Accomplished</td>
<td></td>
</tr>
</tbody>
</table>

Appendix H

Page 520
APPENDIX I: User Test Results – Bar Graphs

Explaining the Charts

Each yellow Box represents each time an attribute is cited for a problem by 1 of the examiners. Each number in the yellow box corresponds to the severity (1 for critical to 4 for minor problem) assigned to that attribute.

Pilot Study Results

OSD Tasks

Attributes and Severity for Problem 1 OSD:

OSD Problem 2: Stuck in VCR mode

OSD Problem 3: Clock Not Set
VCR Tasks

Task 3: VCR Panel: Problem 1 Can't Undo

Task 3: VCR Panel: Problem 2 Must Cancel to Correct

Task 3: VCR Panel: Problem 3

Task 3: VCR Panel: Problem 4:

Task 3: VCR Panel: Problem 5:
Remote Control Tasks

Remote Study Problem 1: Can’t Do Thursday Friday

Remote Study Problem 2: Scanner On Off

Remote Study Problem 3: Can’t Key in Saturday

Remote Study Problem 4: Can’t Understand Process

Remote Study Problem 5: Stuck in Timer Record mode
Bar Code Results

Barcode Study Problem 1: Can’t move up down with remote

Barcode Study Problem 2: Can’t Enter Thursday Friday

Barcode Study Problem 3: Can’t Enter Thursday Friday

Barcode Study Problem 4: Cancel to Correct

Barcode Study Problem 5: Too Slow to Scan

Barcode Study Problem 6: Can’t Scan Backwards
### Barcode Study Problem 7: Can’t Check the Codes

<table>
<thead>
<tr>
<th></th>
<th>CC</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>PM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Barcode Study Problem 8: Can’t Find Scanning Light

<table>
<thead>
<tr>
<th></th>
<th>CC</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>PM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Page 526
Main Study Results

NEC tasks

NEC: Task 3: Problem 1: Navigation Problems

NEC Task 3: Problem 2: Can't Put the Phone Number In

NEC Problem 1: Navigation Problems

Task 4: NEC Problem 2: Multitasking
NEC: Task 5: Problem 1: Web Services

NEC: Task 5: Problem 2: No coverage

NEC: Task 5: Problem 4: Can't access through menu

NEC: Task 5: Problem 5: Navigating

NEC: Task 6: Problem 1: Too complicated

Task 6: Problem 2: Must cancel to correct [no coverage?]

Appendix I

NEC: Task 7: Problem 1: Can’t exit Download Game

NEC: Task 7: Problem 2: Button difficulties

NEC: Task 7: Problem 3: Navigating to Multimedia

NEC Task 8: Problem 1: Slow navigation

NEC Task 8: Problem 2: Identifying Keys

NEC Task 10: Problem 1: Can’t Exit Appointment Creation

Page 529
Appendix 1

NEC Task 10: Problem 2: Can't Edit Existing Date

NEC Task 10: Problem 3: Navigating to the Calendar

NEC Task 10: Problem 4: Wrong Button

NEC Task 10: Problem 5: Adding an appointment

NEC Task 10: Problem 7: Day by Day navigation
<table>
<thead>
<tr>
<th>Task</th>
<th>Problem</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEC Task 12</td>
<td>2</td>
<td>Confirming the save</td>
</tr>
<tr>
<td>NEC Task 12</td>
<td>3</td>
<td>The Delete Key</td>
</tr>
<tr>
<td>NEC Task 12</td>
<td>4</td>
<td>Slow Navigation</td>
</tr>
<tr>
<td>NEC Task 13</td>
<td>1</td>
<td>Can’t find calendar</td>
</tr>
<tr>
<td>NEC Task 13</td>
<td>2</td>
<td>Thought it was saved</td>
</tr>
<tr>
<td>NEC Task 13</td>
<td>3</td>
<td>Can’t cancel current message</td>
</tr>
</tbody>
</table>
### Appendix I

#### NEC Task 14: Problem 1: Can't Send Image

<table>
<thead>
<tr>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>FM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

#### NEC Task 14: Problem 2: Lots of Navigation

<table>
<thead>
<tr>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>FM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

#### NEC Task 14: Problem 3: Multi-Task

<table>
<thead>
<tr>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>FM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

#### NEC Task 14: Problem 4: Replace attached Image

<table>
<thead>
<tr>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>FM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

#### NEC Task 14: Problem 5: Unsure new photo

<table>
<thead>
<tr>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>FM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

#### NEC Task 14: Problem 6: Tries Internet, tries Messages

<table>
<thead>
<tr>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>FM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

Page 533
NEC Task 14: Problem 7: Wrong Navigation

NEC Task 14: Problem 8: Trouble with light turning off

NEC Task 15: Problem 1: Why must game connect?

NEC Task 15: Problem 2: Lost?

NEC Task 16: Problem 1: Didn’t save

NEC Task 16: Problem 2: Delete all

NEC Task 16: Problem 3: Manual Reference
NEC Task 17: Problem 1: Open Application

NEC Task 17: Problem 2: Can't Recall File Name

NEC Task 17: Problem 3: Delete Confirmation

NEC Task 18: Problem 1: Saved Message Location

NEC Task 19: Problem 1: "Off" Button Duration

NEC Task 19: Problem 2: Dislike of Phone

NEC Task 19: Problem 3: Closing the Phone
Samsung Tasks

Samsung Task 1: The ON BUTTON Location

Samsung Task 2: Activation Confirmation

Samsung Task 3: Enter PIN

Samsung Task 4: Long Start up Sequence

Samsung Task 2: No Problem

Samsung Task 3: Problem 1: Can't Put Number in
Samsung Task 3: Problem 2: Navigating to Phonebook

Samsung Task 3: Problem 3: Accidentally Start Call

Samsung Task 3: Problem 4: Pre-existing phone number

Samsung Task 3: Problem 5: Add an additional number

Samsung Task 4: Problem 1: Unaware of folder structure

Samsung Task 5: Problem 1: Can't find Web Option in the Menu

Samsung Task 5: Problem 2: The Sideways Menu

Samsung Task 5: Problem 3: No Coverage
Appendix 1

Samsung Task 6: Problem 1: Can’t Find the Number

Samsung Task 6: Problem 2: Can’t find it in contacts

Samsung Task 6: Problem 3: Not sure if it has been rung

Samsung Task 6: problem 4: No problem

Samsung Task 7: Problem 1: Can’t find SMS functions

Samsung Task 7: Problem 2: Is unaware of left/right scrolling menu

Samsung Task 7: Problem 3: Does not understand soft keys
Samsung Task 7: Problem 4: Does not recognise C is the delete button

Samsung Task 8: Problem 1: Can't identify how to enter plus

Samsung Task 8: Problem 2: Unaware of side scrolling menu

Samsung Task 8: Problem 3: Not sure about soft keys

Samsung Task 8: Problem 4: Thinks calculator functions available elsewhere

NEC Task 8: Problem 5: Slow Navigation

Samsung Task 8: Problem 6: Trouble entering equation
Appendix I

Samsung Task 8: Problem 7: Unsure if calculation completed

Samsung Task 9: Problem 1: Voice not accepted

Samsung Task 9: Problem 2: Navigation

Samsung Task 9: Problem 3: Difficult

Samsung Task 9: Problem 4: Can't Hear Voice

Samsung Task 9: Problem 5: Voice Dial, Act, Tag Confusion

Samsung Task 10: Problem 1: Unintentional Save
## Samsung Task 10: Problem 2: Navigation

<table>
<thead>
<tr>
<th>1</th>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>FM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
</table>

## Samsung Task 10: Problem 3: Difficulty making appointment

<table>
<thead>
<tr>
<th>1</th>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>FM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
</table>

## Samsung Task 10: Problem 4: Scrolling to Date

<table>
<thead>
<tr>
<th>1</th>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>FM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
</table>

## Samsung Task 10: Problem 5: Unexpected response

<table>
<thead>
<tr>
<th>1</th>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>FM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
</table>

## Samsung Task 10: Problem 6: Keys too close

<table>
<thead>
<tr>
<th>1</th>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>FM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
</table>

## Samsung Task 10: Problem 7: Not sure about year

<table>
<thead>
<tr>
<th>1</th>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>FM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
</table>

## Samsung Task 10: Problem 8: Not sure if finished

<table>
<thead>
<tr>
<th>1</th>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>FM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
</table>

## Samsung Task 10: Problem 9: Can’t Put the Phone Number In

<table>
<thead>
<tr>
<th>1</th>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>FM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 11: Problem 1: Can't Complete</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task 11: Problem 2: Not certain if Voice Dial Active</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task 11: Problem 3: Manual not helpful</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task 12: Problem 1: Wrong Key</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task 12: Problem 2: Deletion Confusion</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task 12: Problem 3: Navigation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task 12: Problem 4: Slow Scrolling</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task 12: Problem 5: Menu</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task 12: Problem 6: Not sure if deleted</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Appendix 1

**Samsung Task 13: Problem 1:** does not realise message saves...

<table>
<thead>
<tr>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>FM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Samsung Task 13: Problem 2:** Thinks Message is in Presets

<table>
<thead>
<tr>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>FM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Samsung Task 13: Problem 3:** Can't find message

<table>
<thead>
<tr>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>FM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Samsung Task 13: Problem 4:** Trouble Navigating to messages

<table>
<thead>
<tr>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>FM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Samsung Task 13: Problem 5:** Can’t find message

<table>
<thead>
<tr>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>FM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Samsung Task 13: Problem 6:** Thinks it hasn’t saved

<table>
<thead>
<tr>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>FM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Samsung Task 13: Problem 7:** Looks through menu for messages

<table>
<thead>
<tr>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>FM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Samsung Task 13: Problem 8:** Can’t find saved message

<table>
<thead>
<tr>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>FM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

Page 543
<table>
<thead>
<tr>
<th></th>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>PM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Samsung: Task 14: Problem 1: Minor Problems</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>CF</th>
<th>CS</th>
<th>CT</th>
<th>DR</th>
<th>EF</th>
<th>ER</th>
<th>FG</th>
<th>FL</th>
<th>PM</th>
<th>LN</th>
<th>PR</th>
<th>RS</th>
<th>SM</th>
<th>TN</th>
<th>US</th>
<th>VA</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Samsung Task 15: Problem 1: Dissatisfied</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>