CRANFIELD UNIVERSITY



BENNY EKO TJAHJONO

A DEVELOPMENT METHODOLOGY FOR INDUSTRIAL HYPERMEDIA TASK SUPPORT SYSTEMS

SCHOOL OF INDUSTRIAL AND MANUFACTURING SCIENCE

PhD THESIS

ProQuest Number: 10832156

All rights reserved

INFORMATION TO ALL USERS
The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript
and there are missing pages, these will be noted. Also, if material had to be removed,
a note will indicate the deletion.



ProQuest 10832156

Published by ProQuest LLC (2019). Copyright of the Dissertation is held by Cranfield University.

All rights reserved.
This work is protected against unauthorized copying under Title 17, United States Code
Microform Edition © ProQuest LLC.

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

CRANFIELD UNIVERSITY

SCHOOL OF INDUSTRIAL AND MANUFACTURING SCIENCE

PhD THESIS

Academic Year 2002-2003

BENNY EKO TJAHJONO

A development methodology for industrial hypermedia task support systems

Supervisor: Dr Richard Greenough

April 2003

This thesis is submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy

© Cranfield University 2003. All rights reserved. No part of this publication may be reproduced without the written permission of the copyright owner.

Abstract

Workers in today's manufacturing environments have to cope with a greater variety of complex and rapidly changing tasks than ever before. To ensure consistent high quality of products and manufacturing processes, workers require constantly available source of information that serves as a task support and an *aide-memoire* for settings and task procedures. Hypermedia technology has been proven suitable for this purpose.

However, developing industrial hypermedia task support systems (TSS) is different to developing conventional, non-industrial hypermedia applications. This research, therefore, aims to formulate the development methodology that can be used as a guidance to develop industrial hypermedia task support systems. The development methodology, incorporating formal design methods and design heuristics, has been used to develop a number of industrial hypermedia TSS. Both the hypermedia TSS and the methodology have been evaluated through a series of case studies.

The thesis confirms that the success of the hypermedia task support systems is not only dependent on the usability of the system, but also on the quality of the content. Difficult and time consuming updating processes will cause poor accuracy and currency of the content, resulting in the users not being able to get the benefits from using it. The collaborative authoring environment that exists within the development methodology enables the content of the hypermedia task support system to be easily created and maintained, so as to ensure its usability.

Compared to existing methodologies, the proposed methodology is more comprehensive and more suitable for developing industrial hypermedia systems. In addition to the formal approach typically found in 'off-the-shelf' methodologies, the methodology formulated in this research incorporates design heuristics that aim to improve an application's coherence and to reduce users' cognitive overhead, two important requirements for designing the user interface of industrial hypermedia systems.

Acknowledgement

My thanks first and foremost go to my supervisor, Dr Richard Greenough, who has never been tired in giving me advice, critiques and suggestions throughout the research, particularly during the writing up period. I also owe my sincere gratitude to Professor John Kay for opening an opportunity for me to pursue my PhD through a number of industrial bursaries. The endless support from both of you would never be forgotten.

My appreciation also goes to Motorola, UK, particularly Andy Chan, for allowing me carry out the work in the factory. I also thank Richard Whyman, Ceri Davies, Ann Ibbotson and Tara Beatty, who participated in the confirmatory case studies. Without them, the research would not have been completed.

Last but certainly not least, my special thanks to my family, Yoice and Michelle, who have been very patient and supportive. They deserve my time and attention more than they have had.

Contents

Abstract ..... *i*
Acknowledgement ..... *ii*
Contents ..... *iii*
List of figures ..... *vii*
List of tables ..... *ix*
Glossary of terms ..... *x*
1 Introduction *1*
1.1 The need for a performance support system **2**
1.2 The relevance of PSS at the sponsoring company **3**
1.3 Scope of the research **6**
1.4 Structure of the thesis **7**
1.5 Summary **10**
2 Literature review *11*
2.1 Electronic Performance Support Systems (EPSS) **12**
2.2 Interactive Task Support Systems (ITSS) **15**
2.3 Task Support Systems (TSS) **16**
2.4 Enabling technologies for Task Support Systems **17**
2.4.1 Hypertext, Multimedia and Hypermedia 17
2.4.2 World Wide Web (WWW) 19
2.4.3 Computer and Web based trainings 19
2.4.4 Electronic Documentation Systems 19
2.5 Hypermedia Task Support Systems **20**
2.6 Industrial hypermedia TSS **23**
2.7 Previous research in industrial hypermedia TSS **26**
2.7.1 Assembly operations 26
2.7.2 Automated manufacturing 27
2.7.3 Adaptive assembly instructions 28
2.7.4 Adaptive hypermedia for maintenance 28
2.7.5 Quality control 29
2.7.6 Product support 30
2.7.7 Maintenance and diagnostics 30
2.7.8 Digital maintenance manuals 31
2.7.9 Electronic Manufacturing Instruction 32
2.8 The gap in industrial hypermedia TSS literature **35**
2.9 Justification for the research **36**
2.9.1 Development methodology 37
2.9.2 Evaluation of hypermedia applications 38
2.10 Summary **38**

3 Research programme 39

3.1 Aim and objectives 40

3.2 Programme and methodology 40

3.2.1 Stage 1: Review of hypermedia development methodologies 42
3.2.2 Stage 2: Proposed development methodology and success model 42
3.2.3 Stage 3: Evaluation of the development methodology 42
3.2.4 Stage 4: Confirmatory studies to validate the methodology 44

3.3 Summary 44

4 Review of hypermedia development methodology and success model 47

4.1 Hypermedia design methodologies 48

4.1.1 Hypermedia Design Model (HDM) 48
4.1.2 Relationship Management Methodology (RMM) 49
4.1.3 Object-Oriented Hypermedia Design Method (OOHDM) 51
4.1.4 Web Modelling Language (WebML) 52
4.1.5 Other design methodologies 54

4.2 Hypermedia development methodologies 55

4.2.1 Systems Development Life Cycle (SDLC) 55
4.2.2 Hypermedia engineering approach 55
4.2.3 Industrial Hypermedia Development Methodology (IHDM) 57
4.2.4 Hypermedia Flexible Process Modelling (HFPM) 58
4.2.5 Rational Unified Process (RUP) 59

4.3 Comparison of the methodologies 62

4.4 Design rationale for hypermedia TSS 68

4.5 Hypermedia success model 71

4.5.1 DeLone and McLean model 71
4.5.2 Lucas model 72
4.5.3 Technology Acceptance Model (TAM) 73
4.5.4 Task-Technology Fit (TTF) model 75
4.5.5 Hypermedia usability 77

4.6 Testing and evaluation methodology 79

4.7 Summary 80

5 Proposed development methodology and success model 81

5.1 The need for a new development methodology 82

5.2 Proposed development methodology 85

5.2.1 Step 1: Specification 86
5.2.2 Step 2: Detailed design 88
5.2.3 Step 3: Implementation 97
5.2.4 Step 4: Evaluation 98
5.2.5 Step 5: Deployment and maintenance 99

5.3 Success model for industrial hypermedia TSS 104

5.4 Research instruments 105

5.4.1 Perceived ease of use (EOU) 105
5.4.2 Perceived usefulness (U) 106
5.4.3 Intention to use (IU) 106
5.4.4 Task Technology Fit (TTF) 107
5.4.5 Maintainability (M) 109

5.5 Summary 110

6 Case studies 111

6.1 Case 1: ManTIS simulation mode 113
6.1.1 Step 1: Specification 113
6.1.2 Step 2: Detailed design 116
6.1.3 Step 3: Implementation 120
6.1.4 Step 4: Evaluation 121
6.1.5 Step 5: Deployment and maintenance 127

6.2 Case 2: ManTIS reference mode and training mode 128
6.2.1 Step 1: Specification 128
6.2.2 Step 2: Detailed design 132
6.2.3 Step 3: Implementation 139
6.2.4 Step 4: Evaluation 140
6.2.5 Step 5: Deployment and maintenance 142

6.3 Case 3: Positrol help system 143
6.3.1 Step 1: Specification 143
6.3.2 Step 2: Detailed design 146
6.3.3 Step 3: Implementation 149
6.3.4 Step 4: Evaluation 153
6.3.5 Step 5: Deployment and maintenance 158

6.4 Case 4: Multimedia MPS 159
6.4.1 Step 1: Specification 160
6.4.2 Step 2: Detailed design 163
6.4.3 Step 3: Implementation 169
6.4.4 Step 4: Evaluation 175
6.4.5 Step 5: Deployment and maintenance 176

6.5 Analysis 177

6.6 Summary 180

7 Confirmatory Case Studies 181

7.1 Introduction 182

7.2 Company A: high technology vacuum manufacturer 184
7.2.1 The hypermedia application 184
7.2.2 Case analysis 186

7.3 Company B: automotive engine factory 188
7.3.1 The hypermedia application 188
7.3.2 Case analysis 189

7.4 Company C: technical manual producer 191
7.4.1 The hypermedia application 191
7.4.2 Case analysis 192

7.5 Company D: a central service 194
7.5.1 The hypermedia application 194
7.5.2 Case analysis 195

7.6 Cross-case analysis 196
7.6.1 Requirements specification 196
7.6.2 User involvement during design process 197
7.6.3 Formal design methods 197
7.6.4 Reliance on hypermedia developer 197
7.6.5 Content management system 197
7.6.6 Other people to contribute content 198
7.6.7 Document control and content approval 198
7.6.8 Strategy for links maintenance 198
7.6.9 Formal usability evaluation 198
7.6.10 Success measures 199

7.7 Confirmatory analysis of the development methodology 200

7.8 Summary **202**

8 Discussion and conclusions ***203***

8.1 Research findings **204**

8.2 Research findings compared with research objectives **206**
8.2.1 Objective 1: The development of hypermedia TSS 206
8.2.2 Objective 2: Evaluation of the development methodology 207
8.2.3 Objective 3: Applicability of the development methodology 209
8.2.4 Objective 4: Recommendations 209

8.3 Contribution to knowledge **211**

8.4 Limitations of the research **212**
8.4.1 Core technology 213
8.4.2 Validation of the methodology 213
8.4.3 Measurement of success 214

8.5 Future work **214**

References ***216***

APPENDIX A ***227***

APPENDIX B ***232***

APPENDIX C ***241***

List of figures

Figure 1 – Training-Performance curve [after Stolovich, 2000] 3
Figure 2 – Systems test area 4
Figure 3 – Thesis structure 9
Figure 4 – Sequential access vs hypertext 17
Figure 5 – Hypertext and hypermedia 18
Figure 6 – Focus of evaluation 43
Figure 7 – Research programme 45
Figure 8 – Decomposition of Stage 3 and Stage 4 46
Figure 9 – Relationship Management Methodology [Isakowitz et al, 1995] 50
Figure 10 – Object-Oriented Hypermedia Design Methodology (OOHDM) [Schwabe and Rossi, 1995] 51
Figure 11 – WebML design methodology [Ceri et al, 2003] 53
Figure 12 – Industrial Hypermedia Development Methodology [Crowder et al, 2000] 57
Figure 13 – Hypermedia Flexible Process Model [Olsina, 1998] 59
Figure 14 – Iterative development in RUP [Kroll & Kruchten, 2003] 60
Figure 15 – Two dimensions of the RUP [Kroll & Kruchten, 2003] 60
Figure 16 – Screenshot of the RUP 61
Figure 17 – Hypermedia design issues [Thüring et al, 1995] 68
Figure 18 – The hypermedia design activity [Nanard and Nanard, 1995] 70
Figure 19 – IS success model [DeLone and McLean, 1992] 71
Figure 20 – Implementation success model [Lucas, 1978] 73
Figure 21 – Technology Acceptance Model [Davis, 1989] 74
Figure 22 – Task-Technology Fit (TTF) model [Goodhue, 1995] 75
Figure 23 – System acceptability model [Nielsen, 1995] 77
Figure 24 – Proposed development methodology 86
Figure 25 – Step1: Specification 88
Figure 26 – Step 2: Detailed design 92
Figure 27 – Example of an E-R diagram 93
Figure 28 – Example of a slice diagram 94
Figure 29 – Example of a navigation diagram 95
Figure 30 – User interface design using high fidelity prototype 96
Figure 31 – Step 3: Implementation 98
Figure 32 – Step 4: Evaluation 99
Figure 33 – Step 5: Deployment and maintenance 100
Figure 34 – The proposed development methodology 101
Figure 35 – Proposed success model 104
Figure 36 – E-R diagram of ManTIS simulation mode 116
Figure 37 – Slice diagram of ManTIS simulation mode 117
Figure 38 – Navigation diagram of ManTIS simulation mode 118
Figure 39 – User interface of ManTIS Simulation mode 120
Figure 40 – Sample profile: working experience 124
Figure 41 – Sample profile: task familiarity 124
Figure 42 – The effect of task familiarity to perceived ease of use (EOU) 126
Figure 43 – The effect of task familiarity to perceived usefulness (U) 126

Figure 44 – Relationships between constructs 127
Figure 45 – E-R diagram of ManTIS reference mode and training mode 132
Figure 46 – Slice diagram of ManTIS reference mode and training mode 133
Figure 47 – Navigation diagram of ManTIS reference mode and training mode 134
Figure 48 – User interface of ManTIS reference mode 137
Figure 49 – Link to a PDF file 137
Figure 50 – User interface of ManTIS training mode 138
Figure 51 – Direct link to ManTIS reference Mode 138
Figure 52 – Usage profile of ManTIS reference mode and training mode 141
Figure 53 – File and folder structure of reference mode 142
Figure 54 – E-R diagram of Positrol Help System 146
Figure 55 – Navigation diagram of Positrol help system 147
Figure 56 – Relationships between tables 150
Figure 57 – Positrol help system 152
Figure 58 – Administration page 152
Figure 59 – Correlation between constructs in Positrol help system 155
Figure 60 – Structure of MPS 161
Figure 61 – E-R diagram of MM-MPS 163
Figure 62 – Hybrid slice diagram to display an MPS 164
Figure 63 – Navigation diagram of MM-MPS (Main menu) 165
Figure 64 – Navigation diagram of MM-MPS (User sub-menu) 166
Figure 65 – Navigation diagram of MM-MPS (Author sub-menu) 166
Figure 66 – Navigation diagram of MM-MPS (Administrator sub-menu) 167
Figure 67 – User interface design and navigation path 169
Figure 68 – Relationships between tables 170
Figure 69 – Query design to display an MPS 171
Figure 70 – Screenshots of MM-MPS 172
Figure 71 – Authoring a multimedia MPS 173
Figure 72 – Sign off multimedia MPS 174
Figure 73 – Review section 174
Figure 74 – Refined development methodology 178
Figure 75 – Product catalogue 185
Figure 76 – Text and linking pages 186
Figure 77 – Edit and management pages 186
Figure 78 – Multimedia Quality Management System 189
Figure 79 – OMT's Interactive Service Manual 192

List of tables

Table 1 – EPSS definitions 14
Table 2 – Industrial hypermedia TSS vs non-hypermedia applications 25
Table 3 – Previous work in hypermedia TSS 33
Table 4 – Comparison of hypermedia design methodologies 64
Table 5 – Comparison of hypermedia development methodologies 65
Table 6 – Design principles and their relation to cognitive issues [Thüring et al, 1995] 69
Table 7 – The 12 dimensions of Task-Technology Fit (TTF) [Goodhue, 1998] 76
Table 8 – Attributes of hypermedia usability [Nielsen, 1995] 78
Table 9 – Selection criteria for development and design methodologies 84
Table 10 – Aspects of prospective users 87
Table 11 – Design heuristics for hypermedia applications 89
Table 12 – Proposed development methodology 102
Table 13 – Perceived ease of use (EOU) items 106
Table 14 – Perceived usefulness (U) items 106
Table 15 – Intention to use (IU) items 107
Table 16 – Dimensions dropped due to irrelevant context 107
Table 17 – Dimensions measured and their respective definitions 108
Table 18 – Task Technology Fit items 108
Table 19 – Maintainability items 109
Table 20 – Requirements specification for ManTIS simulation mode 115
Table 21 – Design heuristics and how they were incorporated in ManTIS simulation mode 119
Table 22 – Users' perception of ManTIS simulation mode 125
Table 23 – Requirements specification for ManTIS reference mode 130
Table 24 – Requirements specification for ManTIS training mode 131
Table 25 – Design heuristics and how they were incorporated in ManTIS reference mode 135
Table 26 – Design heuristics and how they were incorporated in ManTIS training mode 136
Table 27 – Users' perception of Reference and Training modes 141
Table 28 – Requirements specification for Positrol help system 145
Table 29 – Design heuristics and how they were incorporated in Positrol help system 148
Table 30 – Users' perception of Positrol help system 154
Table 31 – Users' perception of maintainability 157
Table 32 – Requirements specification for MM-MPS 162
Table 33 – Slices and their attributes to display an MPS 165
Table 34 – Design heuristics and how they were incorporated in MM-MPS 168
Table 35 – Features of development methodology adopted by company A 187
Table 36 – Features of development methodology adopted by company B 190
Table 37 – Features of development methodology adopted by company C 193
Table 38 – Features of development methodology adopted by company D 195
Table 39 – Comparison of the features from the case studies 196
Table 40 – Confirmatory analysis of development methodology 201

Glossary of terms

PSS	Performance Support Systems
EPSS	Electronic Performance Support Systems
TSS	Task Support Systems
ITSS	Interactive Task Support Systems
HDM	Hypermedia Design Model
RMM	Relationship Management Methodology
OOHDM	Object-Oriented Hypermedia Design Method
E-R	Entity Relationship
SDLC	Systems Development Life Cycle
RDBMS	Relational Database Management System
TAM	Technology Acceptance Model
TTF	Task-Technology Fit
ManTIS	Manufacturing Test Information System
Positrol	Positive Control
MM-MPS	Multimedia Manufacturing Process Specification
RUP	Rational Unified Process
UML	Unified Modelling Language

1 Introduction

Today's manufacturers are going through unpredictable changes as a result of global competition [Maskell, 1994; Rosenberg, 1995]. For technology-driven industries such as computers and telecommunications, the challenge is even greater as product life cycles become shorter and new technologies emerge rapidly. The global market has also imposed on manufacturers the need to reduce product lead-time as well as the requirement to comply with quality standards. Price competition with low-wage economies means that conventional manufacturing paradigms are no longer adequate to deal with product diversification and customisation [Maskell, 1994]. Computer-controlled equipment and automation have taken over many labour-intensive tasks but, where manual tasks remain, it has become apparent that proficiency in adopting the new methods and technologies would sharpen the competitive advantage.

1.1 The need for a performance support system

Despite the technological advances of the last two decades, human resources, when appropriately managed, still make a large contribution to the achievement of business goals [Shadur *et al*, 1995]. Shop floor management, for instance, could encourage job redesign to enable tasks to be carried out in a more flexible manner. The use of large numbers of relatively unskilled shop-floor workers in 'Taylorism' has now been replaced by a demand for multi-skilled and flexible workers who are able to work with less supervisor scrutiny and are willing to learn new tasks, as well as new working practices, quickly [Frieling *et al*, 1997]. Other emphasis is on flexible specialisation, where the workers are required to have both functional and numerical capability, as well as social and interpersonal skills [Leat, 1998].

In addition, shop floor workers are required to perform a wide-range of tasks and hence to familiarise themselves with a wide-variety of task procedures [Maskell, 1994]. They may be rotated through different jobs with similar levels of skill and their jobs may be enriched with more responsibilities, for example, to attain product quality and to perform scheduled maintenance. They are also encouraged to contribute their suggestions for improvements and to become more autonomous [Suzaki, 1993]. The above demands have created more pressure for those workers who may be less familiar with new technologies and often alienate those with less experience. They may suffer from stress and fatigue caused by *competency overload* [Kasvi *et al*, 2000], resulting in decreased motivation and enthusiasm, which can lead to absenteeism. Consequently, the manufacturer would suffer from productivity and quality problems.

Since workers cannot be expected to manage these demands supported only by their own memory and experience, they need continual training and skill improvement [Kasvi *et al*, 2000]. However, as the jobs vary and become more complicated, extra resources, such as trainers or expert assistants, should be available when needed during job performance. Indeed, provision of these resources could lead to a high overhead cost. Therefore, traditional methods of delivering training directly to the workplace, particularly to a large number of people have become impractical [Hasebrook, 1999].

The availability of a performance support system (PSS), in addition to the traditional training, ensures persistent information provision to enable cost-effective, on-demand training and knowledge delivery directly to the workplace [Gery, 1991; Kasvi *et al*, 2000]. As shown in Figure 1, PSS aims to maintain the skills of the workers following the traditional training, by providing persistent job assistance [Stolovitch, 2000]. It also acts as a refresher, which may be required, for example due to job rotation, so that workers can always consult the system during task execution [McGraw, 1994].

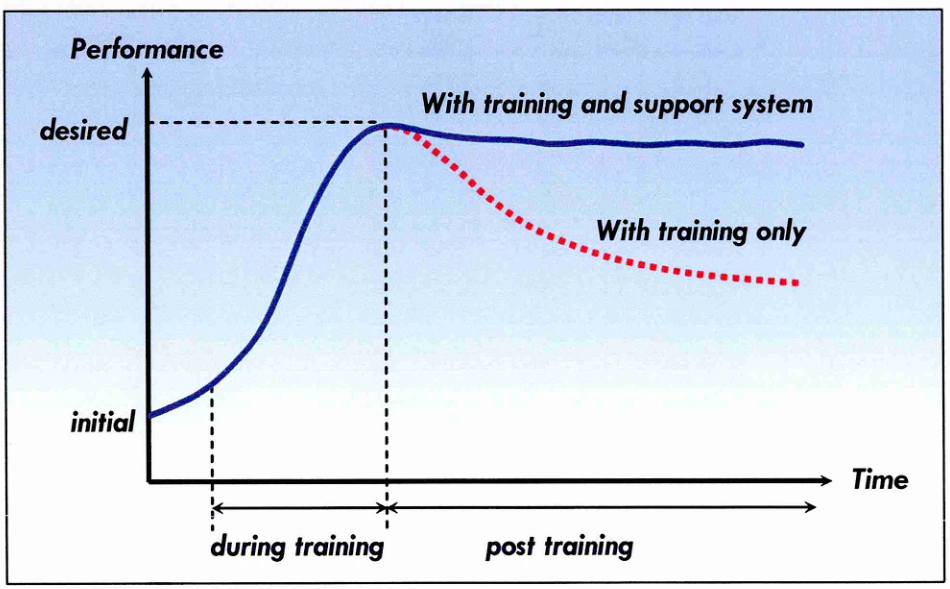


Figure 1 – Training-Performance curve [after Stolovich, 2000]

1.2 The relevance of PSS at the sponsoring company

This research is sponsored by Motorola, a global telecommunication company. Its GSM (Global System for Mobile communication) Product Division facility in Swindon, UK, is dedicated to the research, development and manufacture of the GSM products; the standard for mobile telecommunication technology developed in Europe and adopted in many countries. The manufacturing facility in Swindon involves the production and assembly of, mainly, base transceiver station (BTS) cabinets, simply called base stations hereafter. The base stations are used as the infrastructure to support cellular telecommunication providers all over the world.

The manufacturing facility consists of a number of assembly lines, starting from the printed circuit boards (PCB) components and ending in the systems test area. Due to the pace of developments in electronic fabrication technology, product lead-times are becoming shorter and this has put more pressure on product testing. Because systems test is the last process on the assembly line, a fast and reliable test process is critical to avoid blocking the line and causing late or unreliable deliveries. In order to meet the necessary quality requirements, the base station has to pass a number of tests, carried

out in a computerised test station (shown in Figure 2). The systems test consists of a functional test, which examines whether a base station functions properly, and a protection test, which verifies whether a base station can generate error signals for potential problems such as voltage drops and overheating.

Although the majority of the procedures are under the control of the test software, the overall test process remains labour-intensive. The test software prompts the test operator with instructions to connect the components to the test station, emulating the communication functionality in a cellular network. The successful completion of the test process not only depends on the accuracy of the test equipment, but also on how the operator sets up the connections for the test. Therefore, the competency of the test personnel to carry out the test procedure is crucial.

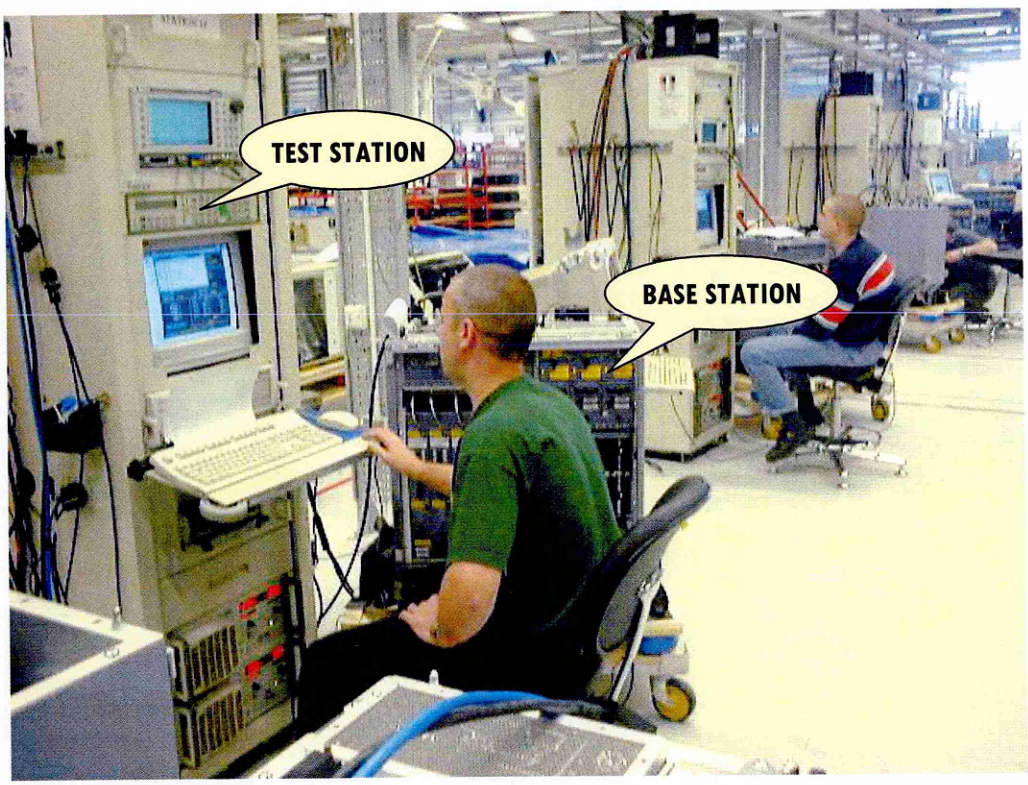


Figure 2 – Systems test area

As each base station is built to customer specifications, the number of different configurations can be very large, resulting in a more complex and a more time-consuming test procedure. To meet production deadlines, the changeover process from one configuration to another should be fast. Such a system is required to retrieve the configuration number, a unique number for each product variant, which is linked to the configuration diagram showing the connection among the product components, and hence to facilitate a faster changeover between product variants.

Before working on the shop floor, new test operators have to be trained by qualified engineers, on a face-to-face basis, following traditional class-based training. With a large variety yet relatively small volume of products, delivering knowledge through this type of training method becomes inefficient and costly. Substantial resources are required to train and to provide continual support to inexperienced operators. Since the new operators are trained on the shop floor, this may generate a greater workload for the engineers, because the demand for training is growing, particularly when introducing new products. This training approach certainly consumes experienced personnel's time that otherwise can be utilised for more appropriate tasks. Assuming the availability of dedicated resources to deliver training materials, the amount of information delivered through a conventional, face-to-face training may be influenced by the social environment and other human factors. This results in variability and inconsistency of the knowledge received by the trainees because different trainers sometimes give the same training material using different methods.

Another issue relates to the characteristics of information available within the company. The complexity of the product, together with the advanced technology involved, has produced abundant information that is generally less structured and hence unsuitable for all levels of expertise. For example, novice operators may find it difficult to search for specific information about the GSM technology, even though it is already covered in most of the manufacturing procedures. Furthermore, technical information about the products, test procedures and test equipment used in the testing area is available only during the training programme, after which the trainee operators will only have limited access to further information. Even if this information was available, it needs to be located as and when required. In many cases, this information is held by the technicians and/or engineers, throughout the company, making it more difficult for novice operators to obtain the required task-related information in a timely manner. The result is that the operators tend to rely on their own memory and past experience, sometimes written on a piece of paper, instead of referring to manufacturing documents.

The format of the technical documents (typically manufacturing processes and procedures) also causes difficulties in extracting the information, due to the presence of cross-references. Most of these documents are in paper-based format and therefore the associated cross-references may need to be presented simultaneously, according to the task in hand. Indeed, from the user's point of view, this is inefficient. Moreover, the process of authoring paper-based documents containing a large number of cross-references requires the document authors to manually embed these cross-references. The authors also need to maintain these references to ensure their accuracy. Therefore, both reading and authoring the documents with persistent cross-references between text, engineering drawings, circuit diagrams and component specifications can be time-consuming and prone to error.

Typical manufacturing procedures, particularly at the shop floor level, have been mostly text-based. Consequently, a large amount of text would be required in order to explain a

long procedural task fully, which can lead to ambiguity and comprehension problems. However, text-based authoring entails a transformation process of complex, unstructured ideas into a structured, linear sequence of text. Therefore, writing a structured and well-understood text-based document can be both difficult and time-consuming.

1.3 Scope of the research

The above are clearly the typical problems a PSS aims to address, and hence the provision of such a PSS with the following features is imperative:

- *Task-centred information system*, where the support documents are restructured in such a way that they provide easy access to the information related to the task at hand,
- *Timely* and *on-demand training*, delivered consistently from trainers to the trainees,
- *Electronic documentation systems,* presented using online writing style, directly accessible from the workplace, and
- *Multimedia objects* that can explain the task more effectively than a large amount of text in paper-based manufacturing documents.

In many cases, the means of providing information to support the task and the presentation of this information, for example manufacturing process documents, are not task-related [Wu *et al*, 1999]. This may be due to the presence of legacy systems, which cause a physical separation between the task and the support documentation. Very often the documents are located elsewhere and are not available when needed. This may prevent support information, such as textual instructions, pictures, or schematic diagrams, from being presented simultaneously, with a consequent reduction in task performance.

Hypermedia technology is suitable for linking various media and presenting support information according to the task in hand. It enables various types of information to be connected to each other and allows the user to navigate through the information in a non-sequential way. Another benefit of using hypermedia is that the links to the support system can be embedded within the task (intrinsic performance support) [Raybould, 2000]. This is not possible with external support, particularly with paper-based documents, where the PSS is physically separated from the task. If hypermedia technology is used for intrinsic performance support, the user will be able to access support information whenever and wherever it is required, since the task and the support system are connected by hyperlinks.

An industrial hypermedia PSS is different from conventional, non-industrial hypermedia applications by its use to support industrial tasks that are typically procedural and require the task performers (factory workers) to interpret the information in a timely manner. An example of such industrial hypermedia PSS is the maintenance information support system. To carry out a repair, a maintenance engineer often has to refer to circuit diagrams, mechanical drawings, parts list and, perhaps, maintenance manuals. At any one time, the engineer requires only a small amount of the information available but needs to cross-reference between several separate documents to find it. With hypermedia, a text may be linked to a drawing and, furthermore, a part of the drawing may also be linked to a video showing how to repair that machine tool.

Kasvi *et al* [1994], Koshy *et al* [1996], Ockerman *et al* [1999] and many others have carried out research in industrial hypermedia PSS aimed at factory workers. They all confirmed that hypermedia is a viable technology for industrial PSS. They also claimed that their applications were successful, as the workers endorsed the applications through a number of usability evaluations adopted mostly from Human-Computer Interaction and Management Information System literature, as well as from hypermedia literature.

Although there are many industrial hypermedia PSS claimed to be successful, as indicated by their acceptance and usage, some of them were failed. The reasons for the failures include management issues and factors such as overly complex authoring and updating processes [Kasvi *et al*, 2000]. Over time, the information became out dated resulting in low usage of the PSS. Unless the systems are being used, the users will not be able to take the full benefits of PSS that are supposed to support their jobs [Kasvi *et al*, 2000].

Developing an industrial PSS based on hypermedia technology is fundamentally different from developing a general purpose hypermedia system [Crowder *et al*, 2003]. In order to ensure that the tasks supported by the PSS are carried out correctly, the content of the PSS has to be supplied by the task experts, who are usually manufacturing or industrial engineers, and not by the hypermedia experts or computer programmers alone [Crowder *et al*, 2000]. The absence of a method that allows people from industry to fully own the PSS may jeopardise the success of such systems [Kasvi *et al*, 2000]. A systematic approach must be taken during the development of the system to ensure that the full benefits of the hypermedia PSS can be achieved. Therefore, formulating the development methodology for successful industrial hypermedia PSS is the main purpose of this research.

1.4 Structure of the thesis

Following the description of background, problems and potential solution identified in this chapter, the next chapters describe the thesis in more detail. The thesis structure is graphically illustrated in Figure 3 and consists of the following sections:

Chapter 2: Literature review.
This chapter reviews the literature underlying the research topics in order to identify the gaps and therefore the need for further research. It discusses the fundamentals of a performance support system and its related technologies, followed by the review of previous research in industrial performance support systems that utilised hypermedia technology.

Chapter 3: Research programme.
In this chapter, the research programme is presented through the development of research aim, objectives and methodology. The research programme itself is split into four individual stages, which are presented subsequently in the next four chapters.

Chapter 4: Review of hypermedia development methodology.
This chapter describes the first stage of the research programme, which is a review of the literature surrounding hypermedia development methodologies, particularly that relating to performance support systems. It also presents various models that may be used to evaluate hypermedia applications.

Chapter 5: Proposed development methodology and success model.
Having reviewed the relevant literature, this chapter proposes the methodology and success model for the successful creation, implementation and maintenance of a hypermedia task support system.

Chapter 6: Case studies.
The chapter describes the case studies conducted at the sponsoring company. The studies aimed to test the proposed development methodology and to evaluate it against the success model. The outcomes of four cases are presented and at the end of each case the methodology will be revised whenever necessary.

Chapter 7: Confirmatory case studies.
This is the last stage of the research programme. Following a number of evaluations of the proposed methodology at the sponsoring company, this chapter presents confirmatory studies to validate the development methodology at other companies across multiple sectors.

Chapter 8: Discussion and conclusions.
The thesis concludes with a discussion of the main research findings that have satisfied the research objectives and confirms that a contribution has been made to the body of knowledge. Finally, it discusses the limitations of the research and recommends the direction for further research.

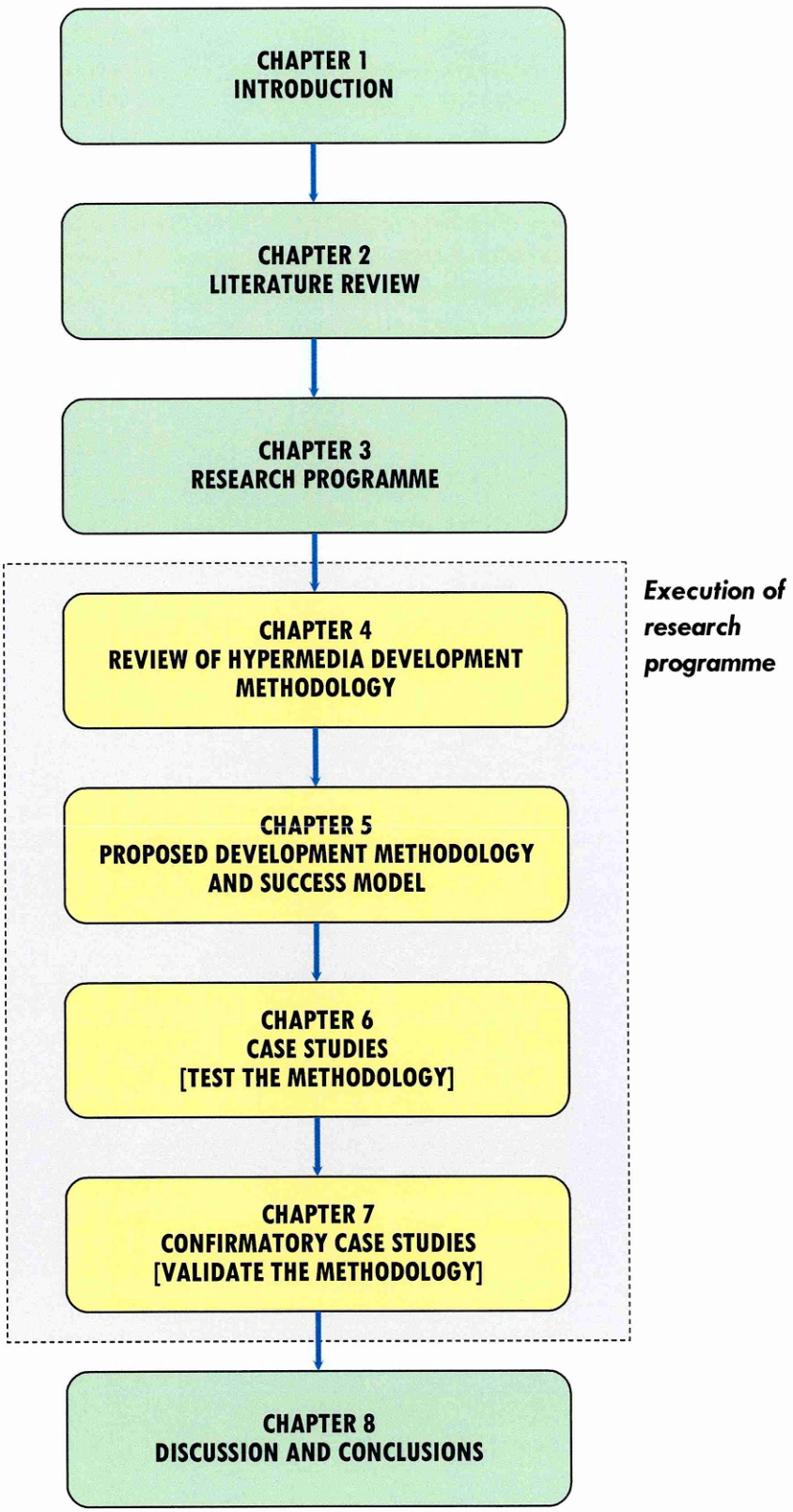


Figure 3 – Thesis structure

1.5 Summary

In today's manufacturing environment where products and hence manufacturing processes are changing frequently, shop floor workers are required to be more multi-skilled. Consequently, they need continual training and support on the job. However, this can be expensive and impractical. A more cost-effective solution is therefore required to provide easy access to the task-related information, delivered electronically and on-demand to support the task performance. Hypermedia performance support systems (PSS) are suitable for the purpose.

Research in industrial hypermedia PSS has been active since mid nineties, along with the advancement of multimedia and Internet technology. Although there were many industrial hypermedia PSS claimed to be successful, some of them were failed. The main reasons were the overly complex authoring and updating processes, resulting in low usage of the systems. Unless the systems are being used, the users will not be able to take the full benefits of PSS.

In order to ensure the success of the PSS, a systematic approach must be taken during the development of the industrial hypermedia PSS to allow people from industry to fully own the systems and to easily contribute the content. The purpose of this research is therefore to formulate a development methodology that can guide industrialists to develop usable and maintainable industrial hypermedia PSS.

2 Literature review

This chapter builds a foundation on which the research is based. It presents a survey of performance support systems and their enabling technologies, followed by the description of previous research in industrial hypermedia. The chapter also discusses the gaps in the literature and issues that have not been resolved as well as the justification for the research.

2.1 Electronic Performance Support Systems (EPSS)

Traditionally, *job aids* are often used to provide information support for less dynamic and relatively simple procedural tasks [Elsenheimer, 1998]. As a type of performance support system, job aids aim to ensure persistent information delivery and to help a worker in performing a task. They are therefore usually located next to the workplace, to act as a quick reference. The form of job aids may vary from a simple cheat sheet, i.e. workers' own notes, to formal Standard Operating Procedures (SOP). In the Toyota Production System, a *Harigami* or a visual label is often attached to a product to check against its physical property, colour, or shape, and hence it quickly identifies what to do to that product [Monden, 1994].

Job aids are an ideal and low cost alternative to traditional instructional methods, for tasks performed intermittently and for procedures that have to be memorised [Elsenheimer, 1998]. They act as a reminder of what has been taught during a training session as well as indicating how to comply with certain procedural standards. They also provide a short cut to information on how to do the task quickly and facilitate knowledge transfer from the expert to the workers in a more consistent way [Bullock, 1982]. They are important, as they bridge the gap between the demand for cost-effective training and the need for just-in-time training as the job changes [Campbell, 1996].

However, when the job environment is increasingly complex and dynamic, a more constantly available information source, which can deal with fast changing support needs, is required. Traditional job aids are not appropriate when a task may consist of a number of different sub-tasks. Consequently, although the main job aids remain the same to a large extent, they may have different variants of other job aids cross-referenced to one another. A delay may result if the appropriate job aid needs to be selected and obtained for each task [Campbell, 1996].

Another related problem is the presentation format of the job aids. Long procedural tasks necessarily need more text or illustration to communicate processes more accurately. However, a large amount of text might lead to comprehension problems and as such might increase the time needed to actually execute the task. Therefore, job aids work well only when the task, and hence the instructions, are simple [Reynolds and Araya, 1995].

Gloria Gery advocated the provision of a better information resource than simple job aids, i.e. one capable of presenting only the relevant information to support job

performance through extensive use of electronic media. She refers to this concept as an Electronic Performance Support System (EPSS) [Gery, 1991], and defines it as:

> *"an electronic system that provides integrated, on-demand access to information, tools, and methodology to enable learning and to generate a high level of job performance with a minimum of support from other people"*

This definition emphasises the role of a non-human system or tool, known as *Infobase* [Gery, 1991], in increasing workers' performance. Thus, the main principle of the EPSS, according to Gery, is '*delivering the right knowledge at the right time and the right place*'. To enable that, various types of computer software are integrated and delivered directly to the workplace where the job is performed. The goal of EPSS, therefore, is to enable the worker to become productive from day one [Gery, 1991]. In other words, *performing* is more important than *knowing* or *learning* [Cole *et al*, 1997]. A number of EPSS definitions, listed in Table 1, have emerged. Despite their differences, the main goal of an EPSS remains the same, i.e. to enable the workers to achieve required levels of task performance, resulting in higher quality products or services [Gery, 1991; Laffey, 1995].

Table 1 – EPSS definitions

Researcher(s)	*Preferred term*	*Definition*
Clay Carr	Performance Support System (PSS)	Tool that ***utilises a computer and its associated technologies*** to provide just the help a performer needs to do a job, just when a performer needs it and in just the form it is needed [Carr, 1992]
Karen McGraw	Performance Support System (PSS)	Support system that incorporates a number of technologies including ***artificial intelligence, computer-based training and hypertext***, to enhance user performance on an operational system [McGraw, 1994].
Angus Reynolds and Roberto Araya	Performance Support System (PSS)	An integrated computer program that incorporates at least two support ***technologies*** (including expert system, multimedia, hypermedia, embedded animations, database) to help in solving work-related problems [Reynolds and Araya, 1995]
Marc Rosenberg	Performance Technology	Technological innovation to enhance performance ***without necessarily promoting learning*** [Rosenberg, 1995]
Barry Raybould	Performance Support Engineering (PSE)	An electronic infrastructure that captures, stores and distributes individual and corporate knowledge assets throughout an organisation to enable individuals to achieve required levels of performance in the fastest possible time and with ***minimum support from other people*** [Raybould, 1995].
Kevin Cole, Olivier Fischer and Phyllis Saltzman	Just-in-time Knowledge	Enhancing users' performance by providing them with knowledge required by the task they are performing ***at the time*** they are actually performing the task [Cole *et al*, 1997]
George Stevens and Emily Stevens	Electronic Performance Support Systems (EPSS); Integrated Information Systems (IIS)	Computer applications that provide ***training, reference*** and ***expert advice*** on demand at the workplace, which together can improve human performance by improving productivity, quality and customer service [Stevens and Stevens, 1995].

2.2 Interactive Task Support Systems (ITSS)

While Electronic Performance Support Systems (EPSS) are well-acknowledged in the USA, researchers from the Laboratory of Work Psychology and Leadership, at the Helsinki University of Technology in Finland use the term Interactive Task Support Systems (ITSS) [Kasvi *et al*, 1996] to describe EPSS. ITSS was conceived from a project to develop a type of information system that supports light assembly tasks, particularly in a modern production environment.

ITSS was defined as:

> *"a source of information supporting competent, good quality and safe completion of work activities, which is available on-demand in association with the task supported".*

Although both EPSS and ITSS are essentially sources of information to support the task, Kasvi *et al* [2000] apparently do not completely agree with the original EPSS definitions listed in Table 1. They believe that systems merely based on EPSS definitions may actually have an adverse effect on employee well-being and the long-term prosperity of the organisation. In particular, they had two main criticisms of EPSS definitions:

- First, according to Kasvi *et al* [2000] EPSS attempts to minimise human support, by defining EPSS as being a tool that enables learning and generates a high level of job performance with a minimum of support from other people [Raybould, 1995]. By minimising human support or interaction, a support system could demote communication and co-ordination between an organisation's units. A system meant to support work activities must also support communication [Kasvi *et al*, 2000].

- Second, supporting work activities must not mean minimising learning [Kasvi *et al*, 2000] or must not eliminate traditional learning methods, such as face-to-face and class-based training. This supports Clark's assertion that reducing training would lead to a potential deskilling of the worker [Clark, 1992]. However, this clearly contradicts what Rosenberg argued about the performance support system that enhances performance without necessarily promoting learning and creates expertise without necessarily creating an expert [Rosenberg, 1995].

Furthermore, Kasvi *et al* [2000] criticised EPSS as an example of the 'old Taylorism', treating the workers as a source of labour. Instead of minimising training, support systems should enhance individual learning and promote both human and organisational interaction so as to enable the organisation to collect, synthesise and refine information created by its members [Kasvi *et al*, 2000].

2.3 Task Support Systems (TSS)

Despite the fundamental arguments asserted by Kasvi *et al* [2000] to promote ITSS, both EPSS and ITSS aim to support a worker's performance by providing a constantly available source of information that is accessible on-demand. Therefore, a more generic term, **Task Support System (TSS)**, will be used hereafter to represent both EPSS and ITSS. This decision is based on the following arguments:

1. Kasvi *et al* [2000] suggest that the word '*interactive*' in ITSS emphasises that the role of end users is to interactively decide only the *order* in which information is accessed. Later on, they limit the term ITSS to refer to the applications used solely to support industrial tasks, such as assembly operations. A more general term, Information Support System (ISS) is used for non-industrial EPSS [Kasvi *et al*, 2000].

2. In one of her articles, Gloria Gery describes the *intrinsic* nature of support that exists in a task-centred system [Gery, 1995]. Task-centred systems, according to Gery, imply that the support system is embedded within the task, so that performers are not aware that they are actually being supported [Raybould, 2000]. Gery uses Microsoft Office's *wizard* as an example to explain intrinsic support [Gery, 1995]. This is relevant to the need to develop a support system for the sponsoring company that is accessible from where the task is executed.

3. In addition to this, Banerji [1999] extensively uses the word *task* to model performance support systems by adopting Checkland's systems thinking theory. A human activity system consists of people and other resources, organised to accomplish a purpose or a task and mediated by tools [Checkland, 1981]. According to Banerji [1999], the role of task support is best described by the nature of the factors that influence the time required for a person to perform a task. Generally, the total time to perform a task depends on two basic factors: the time to find a method and the time for actual task execution, that is:

$$T_{task} = T_{method} + T_{execute}$$

The aims of a task support system are to minimise the overall task performance time and to reduce operational error, the variability of task performance and, eventually, cost [Banerji, 1999]. To fulfil these, it is necessary to identify which of the factors in the equation has the largest effect in reducing task performance time. Assuming that $T_{execute}$ remains constant, T_{task} is a function of T_{method}. The TSS aims to reduce T_{method}. This is analogous to the lean manufacturing principle of reducing (and possibly eliminating) unnecessary or non-value added activity.

4. Finally, the word *task* is commonly used in usability engineering and user interface design. Task analysis [Kirwan and Ainsworth, 1992; Faulkner, 2000] is one of the most important steps in designing both EPSS and ITSS applications.

2.4 Enabling technologies for Task Support Systems

The EPSS definitions given by McGraw [1994], Reynolds and Araya [1995] and Stevens and Stevens [1995] in Table 1 all focus on computer-related technologies. In particular, Stevens and Stevens [1995] emphasise the use of multimedia and hypermedia technologies, whereas both McGraw [1994] and Reynolds and Araya [1995] consider that expert systems and artificial intelligence are needed to develop such an EPSS.

Four major technologies: hypertext and hypermedia technologies; the World Wide Web (WWW); computer-based (and web-based) training systems; and electronic documentation systems, gathered from the literature on industrial task support systems, are discussed in the following sections.

2.4.1 Hypertext, Multimedia and Hypermedia

The word hypertext was coined by Ted Nelson in 1965. He designed a system called *Xanadu*, a repository publishing system intended to manage individual private versions of papers as well as for publishing [Nelson, 1995]. Hypertext systems provide non-sequential information access as opposed to 'reading traditional text' on paper. The latter constitutes a linear sequence that defines the order in which the text is read.

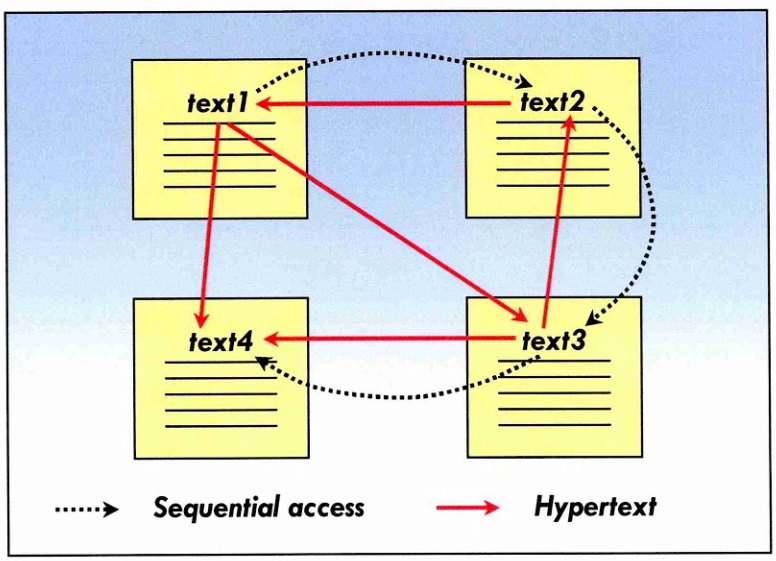


Figure 4 – Sequential access vs hypertext

As shown in Figure 4, with non-sequential access a reader may start reading from *text1* and instead of moving sequentially on to *text2*, he/she can jump to *text3* or *text4*. From there, the reader can also go to *text2* and then back to *text1*. *Text1* to *text4* are known as *nodes* and are interconnected by *links*.

Multimedia is a technology that allows integration of text, images, sound and video to create a dazzling presentation of information [Gunasekaran and Love, 1999]. This integration enables multimedia to convey information with a large degree of comprehension and, therefore, it has been used particularly effectively in computer assisted learning applications, as well as other interactive systems. When a hypertext system includes multimedia, it is also known as *multimedia hypertext* or *hypermedia* [Nielsen, 1995]. In addition, Lowe and Hall [1999] define hypermedia as:

> *"an application that uses associative relationships among information contained within multiple media (multimedia) for the purpose of facilitating access to, and manipulation of, the information encapsulated by the data"* [Lowe and Hall, 1999].

With hypermedia, a reader is not only able to jump from *text1* to *text3*, but also from *text1* to *image* and from *image* to *text2* or vice versa (Figure 5). These interconnections are provided by the associative relationships or links that can be used to navigate within the information space. Examples of hypermedia systems include the World Wide Web (WWW) pages, the Microsoft Windows Help system and Microsoft Encyclopaedia. Hypermedia differs from multimedia in the way in which a user navigates between text, images or videos. In a hypermedia system, a user may decide the path or the sequence to take when a number of options are available and control the duration of the information to be presented [Nielsen, 1995].

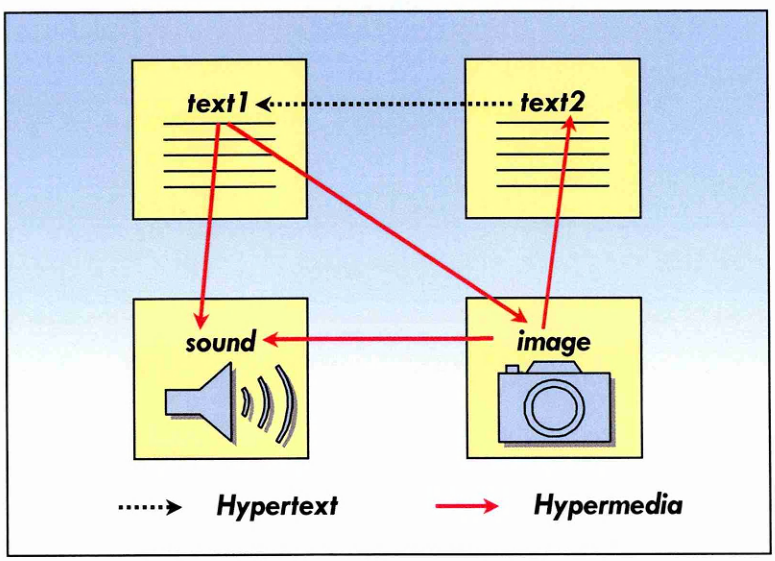


Figure 5 – Hypertext and hypermedia

2.4.2 World Wide Web (WWW)

The fact that the Internet and the web are attracting more users than any other type of Information System has made the web a favourable technology to be used in many applications, including task support systems. Dickelman [2000] reveals that the web has become an important part of the infrastructure of task support systems. Furthermore, he asserts that web-based applications could make some tasks easier to perform as the Internet users become very 'web-savvy'. He points out a number of companies that have already developed web-based task support systems. These companies have also demonstrated how a task can be performed with no training and only a minimal learning curve [Dickelman, 2000]. Duke-Moran *et al* [1999], from IBM, developed a performance support and expert coaching solution. The system uses web technology to retrieve information from a system called DADS98. The supports, i.e. tutorial and context-sensitive, are always available to the users, whenever they are working on the system [Duke-Moran *et al*, 1999].

2.4.3 Computer and Web based trainings

Computer-based training (CBT) is considered to be the backbone of the early EPSS applications. In fact, EPSS and computer-based training are two terms often used interchangeably.

CBT packages were usually developed using *off-the-shelf* multimedia authoring packages such as *Toolbook* or *Authorware* and run on a stand-alone computer. As the number of users increases, the number of stand-alone CBT packages that need deployment also increases. This creates an issue regarding the maintenance and updating of the content. Cross-platform compatibility is also a problem, because multiple versions of the CBT have to be developed for different operating systems. To enable learner-centric CBT, there is also a need to only select the topic the user is interested in from the CBT package. The learner should have a large degree of freedom to jump around and navigate the CBT in non-linear fashion. Traditional CBT usually does not support this [Hall, 1998].

Web-based Training (WBT), on the other hand, adds web features to traditional CBT. As a web system, WBT content is hosted on the web server and is accessible from any web browser regardless of its operating system. This also minimises a lot of maintenance effort, as content updates can only be made from the server.

2.4.4 Electronic Documentation Systems

A task support system in a manufacturing organisation is also a quality control document, created to ensure that a task is carried out consistently and follows the standard. According to BS/ISO 10013, a quality control document must record all processes or procedures and must encourage training of all the members of the company

[BSI, 1995]. Therefore, the documentation system must be up to date, clearly written, well presented, easily accessible and adequately understood [Leung *et al*, 1995; MacMillan, 1997].

There are two fundamental roles of the electronic documentation systems a TSS may perform:

1. Electronic documents may be generated for ***archiving purposes***. This is usually found in an extremely high number of documents where document tracking is crucial [Haywood, 1999]. A numbering system and security features are also required to ensure that no unauthorised access is permitted. Therefore, formal document structure and standard file format are the primary consideration for these types of electronic documents. An electronic or digital manual is an example of such a system. Because the document is archived, structure and file format are the first criteria. SGML and PDF are perhaps the most commonly used file format for this type of electronic document.

2. Electronic documents may serve as ***online help*** and be intended to be displayed directly on visual display units (VDU) or computer screens. Therefore, the user interface and the presentation style of the information are more important to ensure usability. This is understandable since reading text from a VDU is considerably slower (28%) than reading from paper [Hulme, 1984]. BS 7830, a guide to the design and preparation of on-screen documentations, clearly asserts that on-screen documents are fundamentally different from paper documents in that they are inappropriate for sequential reading, page by page [BSI, 1996]. Further research has also revealed that paper documents are unsuitable for direct transfer to an electronic medium [MacMillan, 1996]. Consequently, this type of electronic documentation system must use appropriate writing style and must consider the user interface to ensure usability.

2.5 Hypermedia Task Support Systems

As discussed in Chapter 1, the means of providing information to support the tasks in complex and dynamic manufacturing environment needs to be task-related. The support information itself may include textual instructions, pictures, or schematic diagrams that must be presented simultaneously. Therefore, such non-linear access is desirable to deal with the tasks that need a quick response; otherwise workers may be reluctant to consult the support system.

Hypermedia technology is suitable for these purposes. It enables various types of information to be linked and allows the user to navigate the information in a non-sequential way. An example where hypermedia can be used as a task support system is during maintenance of a machine tool in a factory [Crowder *et al*, 2000]. To repair that machine tool, often a maintenance engineer has to refer to circuit diagrams, mechanical

drawings, parts list and perhaps maintenance manuals. The engineer needs to cross-reference between the various separate documents to find the information required. With hypermedia, the textual information is linked to a drawing and from a part of the drawing the engineer can further access a video showing how to repair that machine through a hyperlink.

In addition to the above, other reasons for using hypermedia as a task support system include:

1. ***Information association***. The hypermedia links (called hyperlinks) allow a human being to partially mimic how a human brain associates pieces of information through non-sequential access [Ginige *et al*, 1995] and uses this relationship to construct understanding of the information. Furthermore, the network structure of information displayed in hypertext or hypermedia represents an organisation that is similar to the associative structure of information in human memory [Marchionini and Schneiderman, 1988]. Browsing the information from a textbook would be an example of this. Instead of reading from the beginning to the end, very often a reader would start from the table of contents, identify the topic of interest, and go to the topic, skipping the other chapters.

   Using the task support system is about building a mental model and understanding how to perform a task. Therefore, the role of a task support system is to enable the user to quickly learn a task. A TSS may have a similar structure to a book. However, the principal difference is that, with hypermedia, the user will be able to locate the support quickly by following the links.

2. ***Embedded support system***. Hypermedia allows the links to be embedded within the task [Raybould, 2000], so it enables direct access to the support system from the point where the task is carried out. The impact is that the task and the support system, which were previously physically separated in the conventional support system, can now be integrated. The big advantage is that it is possible to build an intrinsic (or linked) task support system. An intrinsic performance support system also minimises the time required to locate the support document, as it is embedded within the task. Therefore, theoretically it will minimise T_{method} (see section 2.3)

3. ***Improved comprehension***. Because hypermedia links together multimedia objects. It enables a new way of delivering learning experience to users more effectively [Najjar, 1995]. Multimedia appears to help people build understanding, particularly in the procedural task of operating a system. Mayer and Anderson [1991] found that the use of animation or digital video together with textual description resulted in better understanding than the textual description alone. Moreover, they argued that multimedia could help the learner construct a cognitive model. Kasvi *et al* [2000] also reported that people

supported by multimedia objects (text, pictures and verbal descriptions) completed tasks faster than people who had text only.

The above is consistent with Banerji's equation in section 2.3. The ultimate objective of a task support system is to reduce the time required to find out how to carry out the task (T_{method}). Once the method is available, it has to be understood in order to do the task correctly. T_{method} actually constitutes the time to find the method and the time to understand it. Therefore, the original equation may be modified as:

$$T_{perform} = T_{find_method} + T_{understand_method} + T_{execute}$$

$T_{understand_method}$ is influenced by the format of the task support system. The media used in the TSS must be selected as to minimise $T_{understand_method}$.

4. ***Multimedia can break the language barrier.*** When a task support system is to be used by foreign users, multimedia, in particular animation and illustration, can break the language barrier. Nieminen *et al* [1995] reported that a major part of a complex assembly task could be completed without translating textual information. This is also relevant to what Mayer and Gallini [1990] found.

5. ***Easy authoring.*** Authoring a text-based document implies a transformation of complex, unstructured ideas in the mind of the author into a structured, linear sequence of text [Ginige *et al*, 1995]. Describing a long procedural task, for example maintenance of complex equipment, can be time consuming. Replacing it with a series of illustrations, and possibly video, will speed up the authoring process.

2.6 Industrial hypermedia TSS

The word 'industrial' in industrial hypermedia task support systems in the context of this research implies manufacturing and process industries as opposed to service industries. It refers to the concept of *industrial strength hypermedia* proposed by Malcolm *et al* [1991], who argued that an industrial strength hypermedia system had to evolve beyond the stand alone applications to become a technology that integrates various resources across an engineering enterprise [Crowder *et al*, 2000].

One characteristic that distinguishes industrial hypermedia TSS from *conventional, non-industrial hypermedia applications*[1] is its use to support industrial tasks carried out by shop floor workers [Kasvi *et al*, 1996; Fakun, 2000]. Industrial tasks are typically context-specific, procedural, and they require the task performer to interpret a large amount of information in a timely manner [Hosni & Okrasi, 1992; Kennedy *et al*, 1995]. Examples include assembly instructions, troubleshooting guides and maintenance procedures.

Besides the task, other dimensions such as the users, information content/structure, working environment, computing infrastructure and purpose, may often be used to distinguish industrial hypermedia TSS from non-industrial hypermedia applications. However, it is difficult to identify any fundamental differences between industrial hypermedia TSS and non-industrial hypermedia applications. In practice, the differences are found in the relative importance attached to the features of each type of information system. These differences matter because they affect the design of an industrial hypermedia TSS and mean that one cannot simply transfer good practice from the non-industrial sector or vice-versa.

An industrial hypermedia TSS has to reflect the *mental models* of the users, who are the shop floor workers with typically low to moderate computer experience [Kasvi *et al*, 1996]. Usability is paramount during the development of such systems due to the danger of *disorientation* and *cognitive overhead* [Thüring *et al*, 1995; Koshy *et al*, 1996]. Therefore, although the look and feel may be similar, an industrial hypermedia TSS should be designed using more detail usability criteria than those in non-industrial applications.

The information within an industrial hypermedia TSS is more diverse and more task-related than the non-industrial hypermedia applications. The industrial hypermedia TSS also deals with a lot more different information sources and file formats [Crowder *et al*, 2000], which are supplied or managed by the task experts, i.e. industrial or manufacturing engineers. Although hypermedia allows non-linear access to

[1] Conventional, non-industrial hypermedia applications include web pages for electronic booking of tickets, online library systems, encyclopaedia, e-newspapers, etc.

information, an industrial hypermedia TSS typically requires more sequential navigations.

The working environment where an industrial hypermedia TSS is deployed, may be noisy and dirty, have poor lighting and restricted ergonomics. These lead to additional requirements for more robust computer hardware and software than a non-industrial hypermedia application. In addition, the computing infrastructure in industry usually has more restrictions in terms of access privilege. This is because rapid response to computer breakdown may not be guaranteed during the night shift or at the weekends, which can cause problems when systems are in continuous usage 24 hours a day, 7 days a week.

An industrial hypermedia TSS also acts as a quality document that may need to be audited and may be governed by legislation such as ISO 9000 quality standard [Kasvi *et al*, 1996; Crowder *et al*, 2000]. According to ISO10013-1995, a quality document must provide improved control of working practices for the users of the documentation systems [BSI, 1995]. A non-industrial hypermedia application, on the other hand, is less influenced by such legislation, as it is usually used for entertainment, information retrieval or e-commerce.

Table 2 summarises the typical characteristics of industrial hypermedia TSS in comparison with conventional hypermedia applications.

Table 2 – Industrial hypermedia TSS vs non-hypermedia applications

	Industrial hypermedia TSS	*Non-industrial hypermedia applications*
Task	▪ Complex, procedural industrial tasks ▪ Step by step & interdependent (typically consists of more than 10 steps) ▪ Requires visual inspection and judgement ▪ Maybe time and safety critical ▪ The use of hardware/tool/complex equipment	▪ Less complex, non industrial and typically intellectual tasks ▪ Not necessarily sequential tasks ▪ Less likely to be critical to safety of user
User	▪ Typically shop floor workers with low to moderate computer experience	▪ General users with typically moderate to high computer experience
Information content	▪ High level of diversity ▪ Many different information sources and formats ▪ Industrial task-related ▪ Information supplied or managed by task experts	▪ Less diverse information sources and formats
Information structure	▪ Reflects users' mental model of task to avoid disorientation & cognitive overhead ▪ More sequential navigation, less non-linear access	▪ General usability criteria is sufficient ▪ Fewer restrictions, can be purely non-linear access
Environment	▪ Maybe noisy and dirty ▪ May have poor lighting ▪ Workspace may constraint ergonomics ▪ More pressure in obtaining the information from the TSS	▪ Less disturbances, e.g. in offices, internet cafés ▪ More attention paid to ergonomics ▪ Less hazardous to users' health and less likelihood of contaminant ingress
Infrastructure	▪ Computers with low to moderate spec ▪ Need more robust system due to more specific purpose ▪ Restricted access privilege ▪ May not guarantee 24/7 support	▪ Usually high spec computers ▪ Freedom to configure software/plug-ins ▪ Easy access to support
Purpose	▪ Setup procedure, troubleshooting guides ▪ As quality documents, governed by legislations/standards	▪ Information retrieval, information archival, e-commerce, entertainment ▪ Less influenced by legislation

2.7 Previous research in industrial hypermedia TSS

The following section describes previous work in the use of industrial hypermedia TSS. Although some of the applications are not explicitly called TSS, they aim at assisting the users, mainly shop floor personnel, to carry out their job and hence they inherit many characteristics of the industrial hypermedia TSS listed in Table 2.

2.7.1 Assembly operations

In the late 1980s, researchers from Helsinki University of Technology in Finland investigated the feasibility of using computers to assist training or learning processes. They developed the first prototype of computer-based training called CATS (Computer Assisted Training System) and demonstrated that multimedia is applicable as a training tool. However, experiments carried out at the Nokia Salora TV plant and at Esmi Oy [Kasvi *et al*, 1994] showed that the technology did not permit them to implement such a true multimedia system. Technologies and equipment for authoring such a package were not, at that time, cost effective for industrial applications. Still pictures and sound were also not good enough to support assembly tasks and, moreover, the authoring process was complex.

In early 1993, they developed the second CATS prototype called CATS !! (Computer Assisted Training System II) [Kasvi *et al*, 1994]. Along with the advancement of multimedia technology, CATS !! demonstrated that multimedia could be a viable solution for creating an industrial hypermedia application. CATS !! utilised text, still images and digitised speech to deliver information, particularly training material, to production workers.

The third TSS prototype they developed was called HOTS, and was funded by ABB Industry from 1993 to 1994 [Kasvi *et al*, 1994]. The prototype of HOTS was evaluated in the ABB power electronic plant in Helsinki, to support the assembly of DC converters. The application was considered successful for both training and performance support. Multimedia demonstrated its superiority over paper-based instructions, as it replaced much of the text and the end users preferred to use HOTS.

After HOTS, another multimedia system, called IMS (Interaktiv Montagestøtte) [Kasvi *et al*, 2000], was develop to support the end assembly of the BeoSound 9000, a Bang and Olufsen CD player. Although product variation is quite small, product assembly is very complex. There is a need for TSS at Bang and Olufsen, because of the seasonal nature of high-end audio sales, where more products like CD players are sold in the spring. This seasonal fluctuation often requires the workers to be transferred from other product lines to meet the increased demand and production volumes. The workers must have a basic understanding of their new task, but the IMS was particularly useful as a reminder or when they were uncertain of the correct procedure. The IMS was located next to the workplace and was kept on while staffs were working. Multimedia

documents were available to guide the workers at any stage of the assembly process [Vartiainen *et al*, 1996].

The role of IMS was not limited to task support in the assembly line. During periods when demand was extremely high, temporary assemblers were also recruited. They needed a training and support system that could quickly tell them what to do. A study demonstrated that IMS was well suited for training new workers and, as a result, the need for traditional training was reduced to one quarter, while the rest was replaced by *learning by practice* using IMS [Kasvi *et al*, 2000].

The results from HOTS and IMS prototypes were utilised by Brainware Oy, a Finish software house, to develop a commercially viable information support system called Task Supporter [Kasvi *et al*, 2000]. To a large extent, managing information content in Task Supporter was considered to be easier, when compared with IMS. The multimedia objects, pictures and sounds can be reused in new documents, as well as in other parts of the Task Supporter documents.

2.7.2 Automated manufacturing

Researchers from Clemson University in the USA developed a hypertext system to reduce set-up time and to ensure consistency of set-up procedures at Ryobi Motor Products in South Carolina [Kennedy *et al*, 1995; Koshy *et al*, 1996]. The set-up and calibration process of balancing machines at Ryobi required a number of steps to be carried out by expert technicians. Due to the complexity of the task, both testing equipment and the machine's manual were required. In the absence of the technician, the whole process might be delayed, as the time was spent (mostly by inexperience operators) looking for technical information in the paper-based manual.

To deal with the problem, training was given to the operators in setting up and calibrating the machines, using the manual provided by manufacturer. However, since the structure of the manual was sequential, references to pictures or tables were given using numbers. While reading the step-by-step instructions, the operator had to follow the cross-references to figures or tables. This process was tedious, because the operator might not be able to develop associations between important pieces of information, (text and images) and a lot of effort was required to understand or to integrate necessary information from different pages. Another problem was the difference in the level of knowledge among the operators using the manual. Experience operators might need to skip parts of the manual and go straight to the ones that were needed, but they had to search through the manual in the same way as the novice operators every time the task was performed.

A computer-based system was then developed using a multimedia authoring package called *Toolbook*. The system was essentially a digital version of the paper manual provided by the manufacturer of the balancing machine. It consisted of basic

information about the balancing machine, balancing procedures, machine calibrations and maintenance. The system utilised the full features of hypertext/hypermedia, such as navigation buttons to go back or forward and to go to the main page. By using this system, operators were able to jump between various sections on the digital manual by clicking on the section names. 'Clickable' *hot words* were also embedded in the pictures to describe additional textual information about a particular component.

A usability study was carried out to compare the effectiveness of the hypermedia manual with the paper version of the same manual. The result indicated that, although the operators had only moderate computer experience, they were able to operate the system with minimal training [Koshy *et al*, 1996]. By providing non-sequential navigation, the hypermedia manual improved understanding of the task procedure and substantially reduced the time to perform the calibration. This was because the hypermedia manual had decreased information retrieval time.

2.7.3 Adaptive assembly instructions

An adaptive assembly instruction, called the APPLE A system, was developed by researchers from Heriot-Watt University in the UK as part of an EPSRC project, in collaboration with three leading UK manufacturing companies. The companies produce high-specification, low-volume avionics and aerospace products. The APPLE A system was aimed at assisting factory workers to understand the instructions on how to assemble a product [Rea *et al*, 1998]. This information was traditionally presented in the form of a paper document, comprising a large number of pages. During the life cycle of a product, these documents were frequently updated and reissued due to product changes. The operators received the same assembly documents, irrespective of their level of experience, skill and expertise. It was felt that the provision of a hypermedia task support system to display the assembly plan overcame many problems associated with the currency of paper documents, the needs of each individual operator and the nature of tasks that were difficult to describe using static pictures and text.

The system provided concise, step-by-step instructions on how to complete the task. More information was given by clicking on the hyperlinks. Support documents included links to 3D models, lists of components, drawings, work instructions and a search engine. The APPLE A also featured a basic adaptive system that could display the profiles of the workers required for that task. Once an assembly plan was published, information about the skills required could be compiled and compared with the profiles of the available operators, in order to assist in job assignment decisions.

2.7.4 Adaptive hypermedia for maintenance

Brusilovsky and Cooper [1999] developed a hypermedia task support system called ADAPTS (Adaptive Diagnostics and Personalised Technical Support), to be used by helicopter maintenance technicians. ADAPTS adjusts the diagnostic strategy to the

technician and the task, and dynamically adapts the sequence of repair procedures, set-ups and tests, based on the technician's response. ADAPTS then generates the information, in response to the steps of that diagnostic process. Key to the ability to adapt diagnostic strategy and provide contextual information is a user model, which characterises each technician [Brusilovsky and Cooper, 1999].

What distinguished ADAPTS from ordinary hypermedia task support was the customisation to each technician's experience, knowledge and preferences. Unlike non-adaptive hypermedia systems, ADAPTS dynamically defined and recommended a route for a technician to select during task execution. In determining a recommended route for troubleshooting, ADAPTS matched what needed to be done and whether the technician had the ability to do it.

The strategy used in ADAPTS was user interface adaptation through a collapsible step-by-step instruction to guide the technician through a troubleshooting procedure. For example, an instruction that a technician was already familiar with would be collapsed by default. On the other hand, an inexperienced technician would automatically receive an expanded view of the instruction, showing the detail of the instruction. However, the final decision as to how the information was accessed would be at the technician's discretion, thus giving greater flexibility for navigation.

ADAPTS was first developed as a proof of concept research project, using operational technical data from a Navy H-60 helicopter. The focus was on the integration of condition based maintenance software with adaptive user interface techniques, and on experimentation with user modelling [Brusilovsky and Cooper, 2002].

2.7.5 Quality control

A multimedia TSS was developed by researchers at the Georgia Institute of Technology in the USA, utilising wearable computer devices to improve workers' performance throughout the factory. The research project was called the Factory Automation Support Technology system (FAST) [Ockerman *et al*, 1999]. By using FAST, workers were able to retrieve task-oriented information, such as drawings, procedures, checklists and animation, as well as expert assistance, through a wearable computer connected to the server via a wireless network link. These devices allowed factory workers, for instance quality assurance inspectors, technicians and production supervisors, to receive task-related information even if they were not sitting in front of a computer.

TSS components used in the FAST system were: reference material, task-specific training, job advice and expert performance support. Every time a user needed information from a reference manual, for example quality data, it was readily available through FAST, so there was no need to refer to the actual manual, which was located elsewhere. FAST also acted as a refresher to previously given training materials, so the user could access procedures on how to perform the task. As an expert advisor, FAST

included lessons learnt from experts to help a user carry out a similar task. The expert performance support was perhaps the most sophisticated feature of FAST, as it provided advice to the user on when to consult the system and where to get support.

To investigate the feasibility of the FAST system in industry, two prototypes, namely QALive and WaRP, were developed and tested in industrial plants [Ockerman *et al*, 1999]. QALive was designed to assist quality assurance inspectors in entering quality data directly into the factory's main computer system. The technical data was captured by voice activated input devices while the inspector's hands were occupied in examining the products. FAST provided the inspectors with information, such as how to carry out the inspection and the corrective action required when the standards were not met.

2.7.6 Product support

As part of a European project, researchers from the Manufacturing Engineering Centre at the University of Cardiff proposed a new approach to integrating product information and knowledge into an intelligent product support system known as Intelligent Product Manual (IPM) [Pham *et al*, 1999]. IPM makes use of hypermedia, 3D modelling, expert systems and virtual reality to deliver product-related information. The IPM concept is defined as a computerised interactive product support system that uses expert knowledge and hypermedia to provide just-in-time support to the user during the life cycle of the product [Pham *et al*, 1999]. As it incorporates the above features, IPM can clearly also be categorised as an industrial TSS.

To demonstrate the concept of IPM, an application called ProManual was created to assist engineers and technicians in the maintenance of forklifts. With ProManual, fault diagnosis is made easier and more accurate through the provision of expert knowledge to the user. When a fault occurs, a technician can consult ProManual and step-by-step guidance is given to the technician, to check every possible factor that might cause the problem. The strength of ProManual is the integration of a specialised product design tool called ProEngineer, and the knowledge-based expert system.

2.7.7 Maintenance and diagnostics

The work of Southampton University's Multimedia Research Group in the UK focused on the development of maintenance support applications using an *open hypermedia system* called Microcosm [Hall, 1995]. It was argued that the open hypermedia system was a more complete and flexible information resource because it permitted linking between documents in most native formats, and the link information was stored separately [Crowder *et al*, 1999]. In contrast, in a closed hypermedia system, the link information was embedded within the documents, resulting in more effort required for links maintenance.

Microcosm was used to author an industrial application known as Factory Information Resource Management (FIRM) for a project in collaboration with Pirelli Cables [Crowder *et al*, 2000]. The application was developed for the company's super tension sheathing line, based on the original documentation provided by Pirelli's suppliers and contractors. The information included paper documents, such as drawings in various sizes, text files and a number of reference documents for the programmable controller.

Although the early version of FIRM had shown the capability to hold a number of digital documents, it did not contain support information for the operators to carry out the task [Crowder *et al*, 2000], and as such it was not considered to be a true task support system. However, a proof of concept was demonstrated by connecting a FIRM system with the sheathing line machine, so that the FIRM system could take measurements from the line, allow faultfinding and obtain the required information at the correct location on the system. This would be a great help during maintenance or diagnostics procedures.

2.7.8 Digital maintenance manuals

Research in industrial hypermedia has been carried out by the Manufacturing Systems Department at Cranfield University, through the development of multimedia information systems, particularly digital manuals for the maintenance of machining lines. A number of pilot applications were developed and tested in Ford's engine plant at Bridgend in South Wales.

Initial analysis of interviews with experienced workers showed that during machine downtime, significant time was spent in searching for information rather than fault diagnosis or component replacement [Greenough *et al*, 2001]. To identify a faulty component, a maintenance technician would need to remove the part to see its manufacturing part number or refer to the engineering drawings. Having identified the part, the technician needed to find the stores reference number, in order to check whether the replacement was available or whether one must be bought.

The digital manual enabled Ford engineers to quickly identify the components that needed replacement and consequently helped them find the relevant stores reference number. To enable this, a top-down approach was used to drill-down information about smaller components, from the machine level to sub-assembly level and, finally, to component or part level. Therefore, the approach taken in the proposed digital manual was the one that most IETM designers used, i.e. to link digitised engineering drawings, pictures and text using hotspots or hyperlinks [Greenough *et al*, 2001].

Although Microcosm was chosen to develop the first version, Adobe Acrobat was used to create subsequent versions of the manual. The backbone of the digital manual was Adobe Portable Document Format (PDF) format. The selection was based on several considerations: the viewer was free, there was clear presentation of large size

engineering drawings on the computer monitor, it was easy to author, searching was easy and, more importantly, simple hyperlinks could be established between PDF documents, using hotspots.

2.7.9 Electronic Manufacturing Instruction

Following the success of the digital manuals for Ford, Greenough *et al* [2000] developed another application for Ericsson (in the UK), called the Electronic Manufacturing Instruction (EMI). The EMI had three roles: as an operator's memory extension during the assembly work, as a quality control document that ensured the standard process of assembling mobile phones was followed, and as a training aid for new recruits [Greenough *et al*, 2000]. Ericsson manufactured mobile phones using a combination of permanent and temporary operators. Similar to the Kasvi's Bang and Olufsen case, during the high season more temporary workers were required to assemble the mobile phones. The standard way of training new recruits was through class-based, face-to-face training for a couple of days before they were allowed to start working. This approach, however, was not adequate as the demand for the product was extremely high during the high season.

The first pilot EMI was developed using Adobe Acrobat only. To create a training tool, the paper-based assembly instruction was restructured and divided into a number of steps. Hotspots were used to link one step to another, providing the trainees with more control of learning pace. Long textual explanations on paper were replaced with short, concise sentences. This ensured the usability of the application, as the users would read the EMI from a computer screen. To improve comprehension, digital video or 3D animations were embedded in each step.

Even though the pilot EMI was never fully implemented in the factory, the results from an evaluation, conducted at Ericsson's manufacturing facility near Sheffield, clearly showed that the approach was cost effective and viable for developing such an industrial hypermedia TSS. More importantly, Greenough *et al* [2000] reported that operators were able to navigate EMI. The operators also considered the application both easy to use and useful.

Table 3 summarises the literature review of previous work in industrial hypermedia task support systems.

Table 3 – Previous work in hypermedia TSS

Case	*Tasks to deal with*	*Purpose*	*Enabling technology*	*Success measures*	*Development method*	*Findings*
CATS and CATS!! [Kasvi *et al*, 1994]	Assembly operations [television]	Training of new operators	Multimedia	Not described	Not described	Multimedia can be used as a training tool in an industrial environment
HOTS [Kasvi *et al*, 1994]	Assembly operations [DC converter]	Training and performance support	Multimedia	End user usability	Not described	Multimedia is better than paper-based instructions. Users prefer to use HOTS than the conventional system
IMS [Kasvi *et al*, 2000]	Assembly operations [CD player]	Task support, memory joggers, training	Multimedia	End user usability	Not described	Reduced training cost
Task Supporter [Kasvi *et al*, 2000]	Light assembly	Task support	Multimedia	End user usability, updateability	Not described	**Failure** of the system caused by inability of the industry to update the content easily
Ryobi system [Kennedy *et al*, 1995; Koshy *et al*, 1996]	Machine set up & calibration [balancing motor armature]	Task support, digital manual	Hypermedia	End user usability	Not described	Hypermedia is a viable technology for task support. Shop floor operators endorsed the hypermedia system
APPLE A [Rea *et al*, 1998]	Assembly planning	Task support, decision making	Hypermedia, web	Not described	Not described	Overcome problems associated with paper documents; matching assembly job and skill availability

Table 3 – Previous work in hypermedia TSS [*continued*]

Case	*Tasks to deal with*	*Purpose*	*Enabling technology*	*Success measures*	*Development method*	*Findings*
ADAPTS [Brusilovsky and Cooper, 1999]	Maintenance [helicopter]	Task support	Adaptive hypermedia, user modelling	Not described	Not described	Integrated adaptive hypermedia with condition-based maintenance
QALive [Ockerman *et al*, 1999]	Quality Assurance (QA)	Assists QA engineers to inspect the products; training refresher	Hypermedia, wearable devices	End user usability	Not described	QALive acts as expert advisor and expert performance support. Improved task performance
IPM [Pham *et al*, 1999]	Maintenance & diagnostics [fork lift]	Product support, troubleshooting guides	Hypermedia, web, expert system	Not described	Not described	Product information and diagnostic into an integrated, intelligent product support
FIRM [Crowder *et al*, 2000]	Maintenance & diagnostics [cable sheathing machine]	Information management	Open hypermedia	End user usability	IHDM. MHA	Feasibility of open hypermedia in industry; linking various types of documents in their native format
Digital manual [Greenough *et al*, 2001]	Maintenance [engine machining line]	Task support	Hypermedia	End user usability	HCI + design rationale	Creation of cost-effective hypermedia maintenance manuals using PDF from the legacy documents
EMI [Greenough *et al*, 2000]	Assembly operations [mobile phones]	Task support, training	Hypermedia, web	End user usability	HCI + design rationale	Well accepted by end users who liked use of video and computer animation to illustrate assembly steps

2.8 The gap in industrial hypermedia TSS literature

Previous work on industrial hypermedia task support systems has demonstrated the feasibility of applying them to functions carried out by shop floor workers in industry. The majority of the industrial hypermedia applications reviewed were considered successful, as the end users, particularly the shop floor workers, endorsed their use of hypermedia applications in their workplace. More importantly, they were able to use the applications without adding extra work load. However, apart from Kasvi *et al* [2000], there was no confirmation as to whether the hypermedia task support systems were in use and continually used. The vast majority of the applications reviewed were prototypes and therefore many issues, particularly related to implementation and maintenance, might have been overlooked at the time the evaluations took place.

Unlike prototypes, complete hypermedia task support systems involve design, evaluation, implementation and, more importantly, maintenance of the systems. Consequently, the success factors of a hypermedia TSS should be deduced from complete applications, not from prototypes. In addition, they should be investigated over a long period of time and evaluated. The issues, such as process deployment and maintenance, may become apparent when the prototypes are actually deployed. Therefore, the success measures derived from prototypes may not truly describe the success of the hypermedia application.

It is interesting to note that only one case in the literature reported the failure of a hypermedia task support system. Having successfully developed and licensed *Task Supporter* in a number of manufacturing companies, the software developers reported it as having failed [Kasvi *et al*, 2000]. Although many of the problems were associated with the lack of solid co-operation between the developers and the end users of the application, the main problem was overly complex content maintenance and updates. Furthermore, the application was not designed to be easily updateable by in-house personnel. Nevertheless, what has been reported are merely the reasons why the application failed, not how to ensure that the applications could be successful in the future.

This failure, as pointed out by Kasvi *et al* [2000], also implies that the determinant of success for hypermedia applications which include task support systems is not just the end users usability. Indeed, usability is paramount, but merely conducting the evaluation with the end users appears to be insufficient to claim success. It has been demonstrated that a hypermedia application which claimed to be successful in the beginning could fail in the longer term if content maintenance was too complex. Therefore, in addition to end user usability criteria, other 'aspects' that might also contribute to the success of the hypermedia task support system applications need to be investigated in order to claim success.

Finally, apart from Crowder *et al* [2000], the researcher has located no confirmation from other cases as to whether they have adopted formal hypermedia development methods (excluding the design rationale for user interface), during the development of their hypermedia applications. Many researchers (for example Murugesan *et al* [2001], Lowe and Hall [1999], Powell *et al* [1998]) have pointed out that the formal methods are necessary to help in the creation of structured hypermedia applications and hence reduce effort in maintenance and update.

2.9 Justification for the research

Developing an industrial hypermedia TSS is fundamentally different to developing ordinary hypermedia systems. The development of hypermedia TSS requires close collaboration between manufacturing process experts, who are familiar with the tasks, and hypermedia developers [Vartiainen *et al*, 1996; Ginige and Murugesan, 2001]. To ensure that the tasks supported by the TSS are carried out correctly, the content of the TSS has to be supplied by the manufacturing process experts [Kasvi *et al*, 2000; Crowder *et al*, 2000]. Reliance on hypermedia developers alone to collect, integrate and manage the TSS contents from manufacturing process experts will add considerable time delay in authoring and updating the hypermedia TSS. This in turn will reduce the currency and accuracy of the application. Studies have shown that poor currency and accuracy is one of the reasons for Information System failure [Davis, 1989; Goodhue, 1995], as the users are unable to benefit from using the applications.

On the other hand, giving the responsibility to the manufacturing process experts to author and to generate the contents of hypermedia TSS themselves may increase their workload, as they are required to familiarise themselves with the hypermedia technology. Studies further revealed that failure also occurs due to skill deficit in the upkeep of the application [Kasvi *et al*, 2000]. There is therefore a risk of system failure if non-hypermedia experts are to be involved in updating and maintaining the hypermedia applications.

Fakun [2000] asserts that unwillingness of the end users to use the hypermedia system is not the sole cause of information systems failures. Other causes include (1) problems with computers (or other hardware), (2) problems with the application itself, (3) problems during development and (4) problems during implementation of the hypermedia system. Obviously, the last two problems will not have been identified from the prototypes or pilot systems found in the cases described earlier. This is because the prototypes usually cover only a small set of manufacturing tasks and are used for relatively short periods of time. Therefore, it is apparent that the success of hypermedia TSS cannot be solely determined by the usability perceived by the end users.

More importantly, according to Jiang *et al* [1998], the success of any IT system including hypermedia is more likely to be determined by its ability to meet all stakeholders' expectations. In the case of a task support system, the stakeholders

include the *end users* and the *manufacturing process experts* who will contribute the content.

Kasvi *et al* [2000] remind us that the failure of *Task Supporter* was due to skill incompatibility between the external developers (a software house called Brainware Oy) who created the application and the in-house personnel who needed to update the content of the software. Or as Kasvi *et al* [2000] put it:

> "*people who understand the task to be supported by the Task Supporter should maintain the content and ensure that the information is up to date, not people who understand the computers used to support the task*" [Kasvi *et al*, 2000, pp. 216].

Crowder *et al* [2000] put forward a similar argument to Kasvi *et al* [2000] in relation to the authoring function of the hypermedia task support system. They take into account the fact that the people who are likely to implement the hypermedia task support would be from the industrial environment, and not computer programmers. Therefore, the absence of a method that allows the people from industry to wholly own the task support system can also jeopardise the success of the application.

Focusing our attention on the major problems in the development and maintenance of current industrial task support systems, two major aspects that need further investigation are identified in the following sections.

2.9.1 Development methodology

Traditional methods of developing hypermedia applications have been *ad hoc* and not very structured [Balasubramanian and Bashian, 1998; Murugesan *et al*, 2001]. This could lead to inconsistent design, maintenance and updating problems and eventually to failures, particularly when the scale of the TSS applications becomes larger and more complex [Garzotto *et al*, 1993; Brereton *et al*, 1998; Murugesan *et al*, 2001]. Kasvi *et al* [2000] supported this argument by pointing out that although the *Task Supporter* was initially successfully developed and received acceptance from the end users, it eventually failed due to the overly complex authoring and maintenance processes.

If failure is due to these problems, the research should devise a better development methodology that would allow the content of hypermedia task support systems to be easily updateable and maintainable, particularly by manufacturing process experts.

The use of appropriate hypermedia development methodology will consequently enable the hypermedia developers to design, implement and update industrial hypermedia task support systems with less effort [Taylor *et al*, 2002; Bieber *et al*, 1997; Isakowitz *et al*, 1995; Garzotto *et al*, 1995].

2.9.2 Evaluation of hypermedia applications

Having satisfied the above requirements, evaluation must be carried out to assess whether the TSS has met the usability criteria from both the end-users' and the process experts' points of view. This is relevant to what has been proposed by Kasvi *et al* [2000] and Nielsen [1995], in which the usability criteria have to be applied to both end users and the authors of the hypermedia applications.

2.10 Summary

Due to the fact that EPSS [Gery, 1991] and ITSS [Kasvi *et al*, 1994] aim at providing on-demand, task-related information to improve task performance, a more generic term, Task Support System (TSS), is used in this thesis to represent both. Previous work on industrial hypermedia has demonstrated that hypermedia is suitable for task support systems. This is because hypermedia allows users to navigate the system and to search for information in a non-sequential way. It also enables various types of information to be linked to each other, so that task-related information can be presented in a more effective way.

A review of previous work in industrial hypermedia showed that most of the cases did not indicate whether the hypermedia applications were used continuously in the company. Nevertheless, it was claimed that most of the applications were successful, as indicated by user acceptance of the applications. Only Kasvi *et al* [2000] described the failure of their system, which was mainly due to overly complex authoring and maintenance of the hypermedia application.

Developing an industrial hypermedia task support system is different to developing ordinary hypermedia applications. Although the hypermedia developer plays an important part during the design phase, the content has to be contributed by people who understand the task to be supported by the system. The success of the application is therefore not only dependent on the usability of the system, but also on the quality of the content. Difficult and time consuming updating processes may cause poor accuracy and currency of the task support system. If the content is not current, and hence less useful, it could jeopardise the success of the system, because users would not benefit from using it.

3 Research programme

Chapter 2 reviewed the literature related to hypermedia applications for manufacturing task support. The gaps in the literature, as well as the justification for the research, have also been identified. This chapter discusses how the overall research programme will be conducted. It sets out the aim and objectives of the research, followed by the selection of research methodology. It also describes the data collection and analysis methods adopted throughout the research. To fulfil the research objectives, a research programme consisting of four stages is proposed, followed by the detailed description of these stages.

3.1 Aim and objectives

Previous research on hypermedia task support systems (TSS) has considered the success of hypermedia applications from two different viewpoints. Human-Computer Interaction (HCI) literature looks at *usability* from the viewpoint of the end users, often using inferential statistics to deduce the success factors, whereas hypermedia and computer science literature proposes structure, method and the features of the hypermedia applications as the success factors. However, neither of them confirms whether the success criteria are still relevant if a hypermedia TSS is deployed in a large manufacturing organisation, involving a large number of process experts who frequently contribute the content of the TSS.

Therefore, the overall aim of this research is to formulate a development methodology that can be used to guide the industrialists who intend to develop industrial hypermedia task support systems from conception of idea to systems deployment.

To accomplish the above aim, the following research objectives have been identified:

1. To develop examples of industrial hypermedia task support systems that satisfied the expectations of all relevant stakeholders of the systems, based on the proposed methodology

2. To investigate the extent to which the proposed development methodology has supported the development of industrial hypermedia task support systems

3. To evaluate the applicability of the development methodology to manufacturing industry in general

4. To provide recommendations to manufacturing organisations that intend to implement industrial task support systems based on hypermedia technology

3.2 Programme and methodology

This research has been made possible through a project sponsored by Motorola, in the UK, during which the researcher spent two years working closely with shop floor

personnel, including operators, engineers and managers. Motorola is a global telecommunication company whose products and hence manufacturing processes have short life cycles. The availability of the industrial setting allowed the researcher to carry out a detailed observation and inquiry, with the full cooperation of many individuals at different levels within the company.

The project has provided a basis for executing a research programme through the deployment of a number of hypermedia task support systems and evaluation of the success model, during the solution of *real industrial* problems [Robson, 1993]. In addition, the company also allowed the researcher to focus the inquiry on a specific situation and to conduct a number of iterative processes that will be crucial to the success of the research. To enable the researcher to observe and experiment with the factors during the research execution in a more flexible way, the research methodology adopted was case study based [Yin, 1994; Robson, 1993; Bryman, 1989]. Case study research has served various purposes, i.e. it allowed the generation of theory as well as the testing of it, insight was obtained into a previously unknown area in an exploratory manner and findings from other studies were confirmed [Bryman, 1989; Yin, 1994].

To ensure the achievement of the research objectives, a research programme consisting of four stages is proposed:

- **Stage 1: Review of hypermedia development methodologies.** The review and analysis of existing development and evaluation methodologies is imperative to enable the researcher to form the foundation for the proposed methodology.

- **Stage 2: Proposed development methodology and success model.** Both the initial development methodology and the success model are proposed at this stage, arrived at by adopting some features of existing methodologies and success models.

- **Stage 3: Evaluation of the development methodology.** In this stage, the proposed methodology from Stage 2 is tested at the sponsoring company through a series of case studies. At the end of each case study, the methodology is refined and the new methodology with new features is proposed.

- **Stage 4: Confirmatory studies to validate the methodology.** This stage attempts to corroborate the outcomes from the previous stage, by presenting the model to be validated by to other companies.

The following sections describe the research programme in more detail.

3.2.1 Stage 1: Review of hypermedia development methodologies

Such a methodology is required to develop structured and maintainable hypermedia applications. Therefore, a more detailed literature survey is required to review existing hypermedia development methodologies and to choose methodology that is almost suitable to be used to develop the hypermedia applications for the sponsoring company. The chosen development methodology will also become the foundation for the proposed methodology that will be tested and refined further.

This stage also aims to pull together a number of success models from a range of literature on the areas of hypermedia, human-computer interaction and software engineering. These success models will be integrated into a new model that can be used to evaluate the hypermedia task support systems developed using the proposed development methodology.

3.2.2 Stage 2: Proposed development methodology and success model

This stage formulates the hypermedia development methodology and success model for the industrial task support system obtained from Stage 1. The proposed methodology consists of a number of steps representing a full life cycle of the hypermedia task support system, from requirement specification to maintenance. The adoption of both formal and informal techniques provides a more structured method for developing the TSS.

In addition to the methodology, this stage also proposes a success model that will be used to evaluate the TSS. Although, to a certain extent, the model is deduced from existing models in the literature, the underlying principle of the proposed model focuses on the stakeholders of the *development process* of the hypermedia TSS. As pointed out earlier, the stakeholders include the *end users* and the *manufacturing process* experts, who will contribute the content of the TSS. They are usually manufacturing or industrial engineers who understand the tasks very well.

3.2.3 Stage 3: Evaluation of the development methodology

Evaluation and testing of the development methodology will be conducted at the sponsor's site, through a series of case studies. In each case, a hypermedia TSS will be developed using the proposed methodology and will be evaluated against the proposed success model. The findings from each case will become the basis for the refinement of the proposed methodology.

The case study research allows a combination of quantitative and qualitative approaches. One of the advantages of using the two approaches together is that one approach can be used to check the findings from the other [Bryman, 1989]. Another is

that the approaches can complement each other, should one approach be hard or impossible to achieve.

A quantitative approach is likely to produce quantified and, if possible, generalised conclusions drawn from the relationships of one set of facts to another [Bell, 1987]. The case studies will therefore be carried out using a survey and questionnaires. The subjects will be predominantly shop floor operators. The qualitative approach, on the other hand, deals with insight gathered from individuals' perceptions of the subject being observed [Bell, 1987]. This usually requires interviews involving engineers and managers, although shop floor operators might be involved in some circumstances.

In each case study, evaluation is applicable to both the end users and the manufacturing process experts. However, the focus is different in each case. Early cases will focus more on the success criteria from the end users' point of view rather than from that of the manufacturing process experts. As experience builds up, the evaluation will focus more on the manufacturing process experts. This is illustrated in Figure 6.

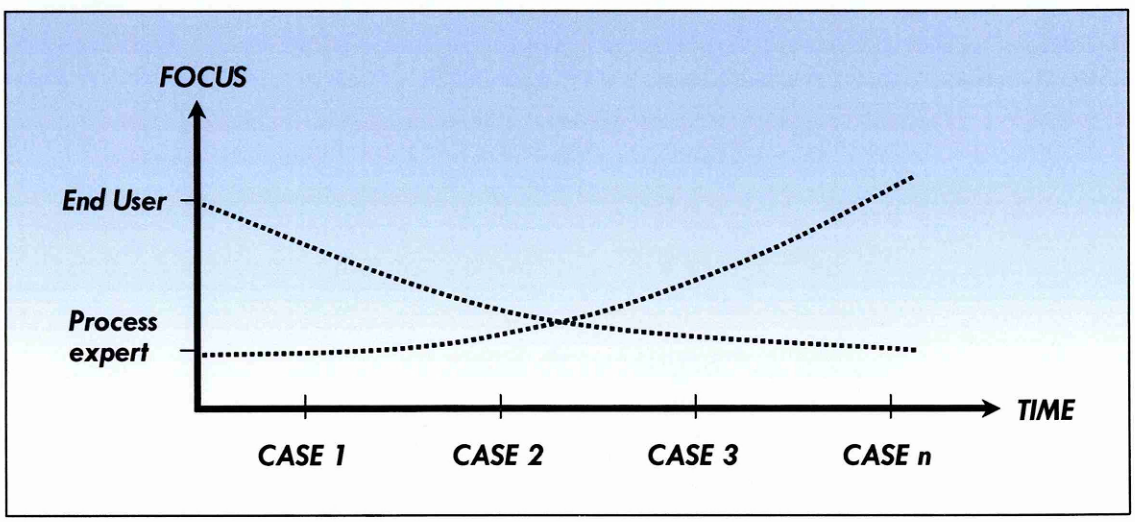


Figure 6 – Focus of evaluation

During the execution of the case studies, the researcher will develop and implement the application, as well as conducting evaluations within the sponsor site, obtaining information from the lessons learnt and finally refining the model. The refined model will be used in further case studies and the development methodology will also be applied in the subsequent hypermedia applications. The model and methodology are once again tested, and refined. Each case study would add to the construction of the development methodology and success model, as described in the research objectives.

The above approach is also known as action-based research, entailing an iterative process of problem solving, which moves from problem identification to data collection,

diagnosis, formulation of action, implementation and evaluation [Bryman, 1989; Coughlan and Coghlan, 2002]. This is commonly characterised by the research setting where the researcher participates actively in the research framework and the research is typically a process of linking theory and practice, for instance, the development of new or changed system development methodologies [Baskerville, 1999].

3.2.4 Stage 4: Confirmatory studies to validate the methodology

Confirmatory case studies involve a number of companies across multiple sectors that have already developed and implemented task support systems based on hypermedia technology. The purpose is to validate the development methodology and the success model formulated in Stage 3. The studies will be carried out using a qualitative approach through semi-structured interviews.

Prior to the validation of the model, each company will be interviewed about various aspects of the development of their hypermedia systems, without first presenting the model to the interviewee. The purpose is to avoid the risk of bias and to isolate the findings that the researcher wishes to validate, which could not be done if the model was introduced to the interviewee at the beginning of the interview. This should ensure that the researcher can objectively compare the developed methodology with that of the company and thus investigate the generality and robustness of the methodology.

The overall research programme is graphically illustrated in Figure 7. The shadowed boxes in both Stage 3 and Stage 4 in Figure 7 indicate that these stages comprise a number of sub-stages, as illustrated in Figure 8.

3.3 Summary

This chapter has set out the research aim and objectives. The availability of an industrial sponsor, allowing the researcher to carry out detailed observation and to experiment with various factors during the research execution in a more flexible way, has led to the selection of the case study-based research methodology. A four-stage research programme has been proposed to ensure the achievement of the research objectives. The first stage reviews the existing methodologies and success models from a range of disciplines, particularly hypermedia engineering, human-computer interaction and Management Information Systems (MIS). This leads to the proposed development methodology and success model that will be tested and refined throughout the case studies. Finally, to ensure its generality, the final methodology will be validated through confirmatory case studies.

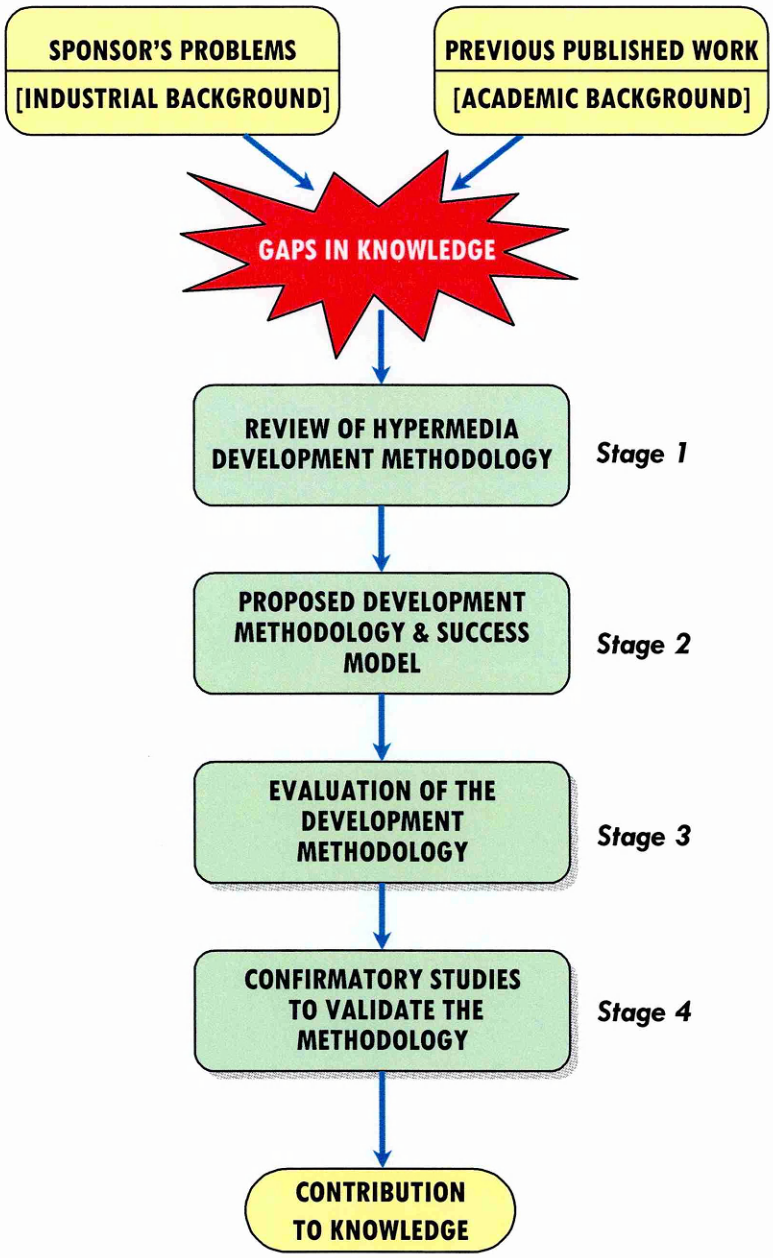


Figure 7 – Research programme

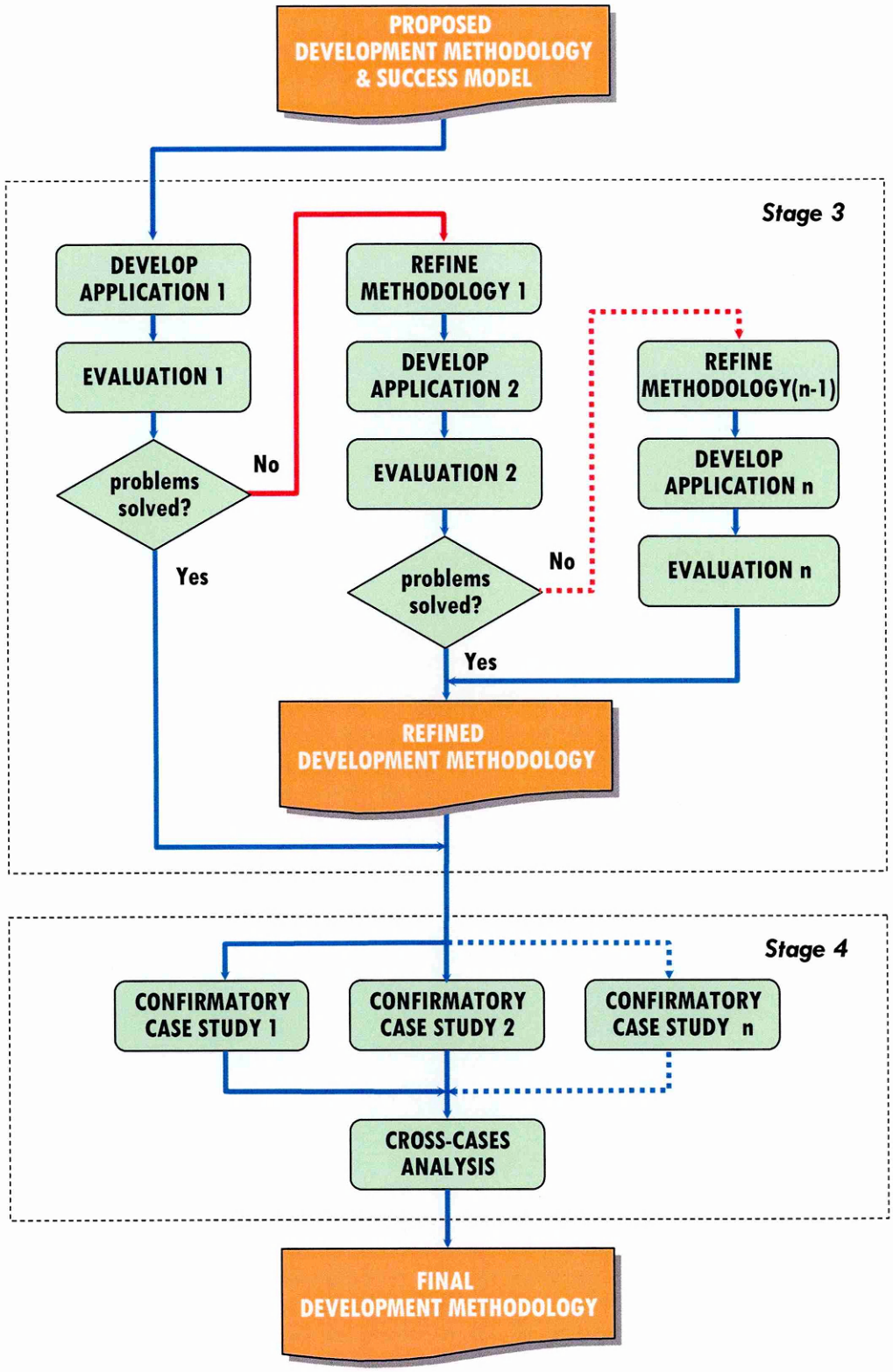


Figure 8 – Decomposition of Stage 3 and Stage 4

4 Review of hypermedia development methodology and success model

This chapter describes the first stage of the research programme. The objective is twofold. First, it aims to survey existing methodologies that are well known and have been widely used to develop hypermedia applications. Second, it also investigates the extent to which the success models from a range of academic literature, including hypermedia, human-computer interaction and management information systems, are suitable for evaluating hypermedia industrial task support systems.

The chapter is organised into six sections. Section 4.1 discusses the ***design*** methodologies and Section 4.2 discusses the ***development*** methodologies, for producing hypermedia applications, explored from hypermedia and software engineering literature. The design methodology includes techniques, tools and approaches used to design the applications. The development methodology, on the other hand, constitutes broader and more comprehensive activities that typically include *design*, implementation, testing and maintenance. In this thesis, both the design methodologies described in Sections 4.1 will be referred to as ***formal design methodologies***. Section 4.3 provides a review on both the design and development methodologies described in previous sections. This is followed by Section 4.4, which presents the design rationale (the so called ***informal design methodologies***) for the hypermedia task support system adopted from research in the area of human-computer interaction and usability engineering. Section 4.5 presents the hypermedia success models and, finally, the evaluation methodologies are discussed in Section 4.6.

4.1 Hypermedia design methodologies

Although there are a relatively large number of techniques and methodologies for hypermedia design, they mostly originated from the Hypermedia Design Model (HDM) [Garzotto *et al*, 1993], and therefore HDM has been a popular reference point among hypermedia researchers [Lang, 2001]. HDM integrates features of Entity-Relationship modelling (E-R) [Chen, 1976] and the Dexter hypertext reference model [Halasz and Schwartz, 1994]. After HDM, various proposed methodologies emerged, although ultimately they evolved from HDM, E-R modelling and Object-Oriented (OO) modelling [Booch, 1994]. The following sections explore the design methodologies in more detail.

4.1.1 Hypermedia Design Model (HDM)

Hypertext Design Model (HDM) is a data model that describes existing, or to be developed, hypertext or hypermedia applications in a system independent manner, without too much regard for implementation detail [Garzotto *et al*, 1993]. HDM considers the hypermedia application on two layers: the *hyperbase layer* and the *access layer* [Coda *et al*, 1998]. The *hyperbase layer* models the information structure that represents the information domain, whereas the access layer provides an entry point for navigation to the information domain.

HDM uses of a number of modelling elements, known as *primitives*, mostly adopted from both the Dexter model [Halasz and Schwartz, 1994] and the E-R model [Chen, 1976]. They are *entities*, *components*, *perspectives*, *units* and *links*. In addition, a *schema* is used to specify the structure of the hypermedia application. HDM also employs another primitive called *outline* to describe the access structure and to locate the most convenient access point at which to start.

Looking at the primitives used, HDM shares some apparent similarities with Entity Relationship (E-R) model, particularly because the term *entity* was adopted from E-R model. However, [Garzotto *et al*, 1993] argue that HDM entities are more structured than the E-R entities, which do not have structural links.

HDM focuses on a systematic, structured development approach that should benefit the hypertext design process, particularly in the case of large and complex applications [Garzotto *et al*, 1993]. The structured approach is concerned with *authoring-in-the-large* as opposed to *authoring-in-the-small*. The former is concerned with the description of overall information elements and navigational structures of complex applications, whereas the latter is concerned with implementation details, such as the nodes and their presentations. Since HDM only provides the framework for analysing the internal structure of a hypermedia application and does not explicitly describe the design methods, it is considered to be a design model rather than a methodology.

4.1.2 Relationship Management Methodology (RMM)

Isakowitz *et al* [1995] proposed a hypermedia development methodology called the Relationship Management Methodology (RMM). RMM evolves the HDM [Garzotto *et al*, 1993] and the Entity-Relationship (E-R) modelling techniques [Chen, 1976] by embedding its hypermedia design concept into a structured methodology. The E-R representation was chosen for a number of reasons. Firstly, the E-R model has been used widely in system analysis. Secondly, it is well documented and, finally, it can model information dependency in numerous application domains.

RMM addresses the design and construction of hypermedia application through a seven-step process (Figure 9). These steps include *entity-relationship design*, *slice design*, *navigation design*, *user interface design*, *protocol conversion design* and *run-time behaviour design*, plus construction and testing. RMM particularly focuses on Steps 1 to 7 within the software development life cycle.

During *E-R design*, entities, attributes and relationships that will become nodes and links in the resulting hypermedia application are identified. *Slice design* involves grouping entity attributes for presentation. Slices are presentation units that appear as pages of the hypermedia applications. *Navigation design* identifies the navigation paths through a number of access primitives: links, grouping or menu, index, guided tour and a combination of index and guided tour.

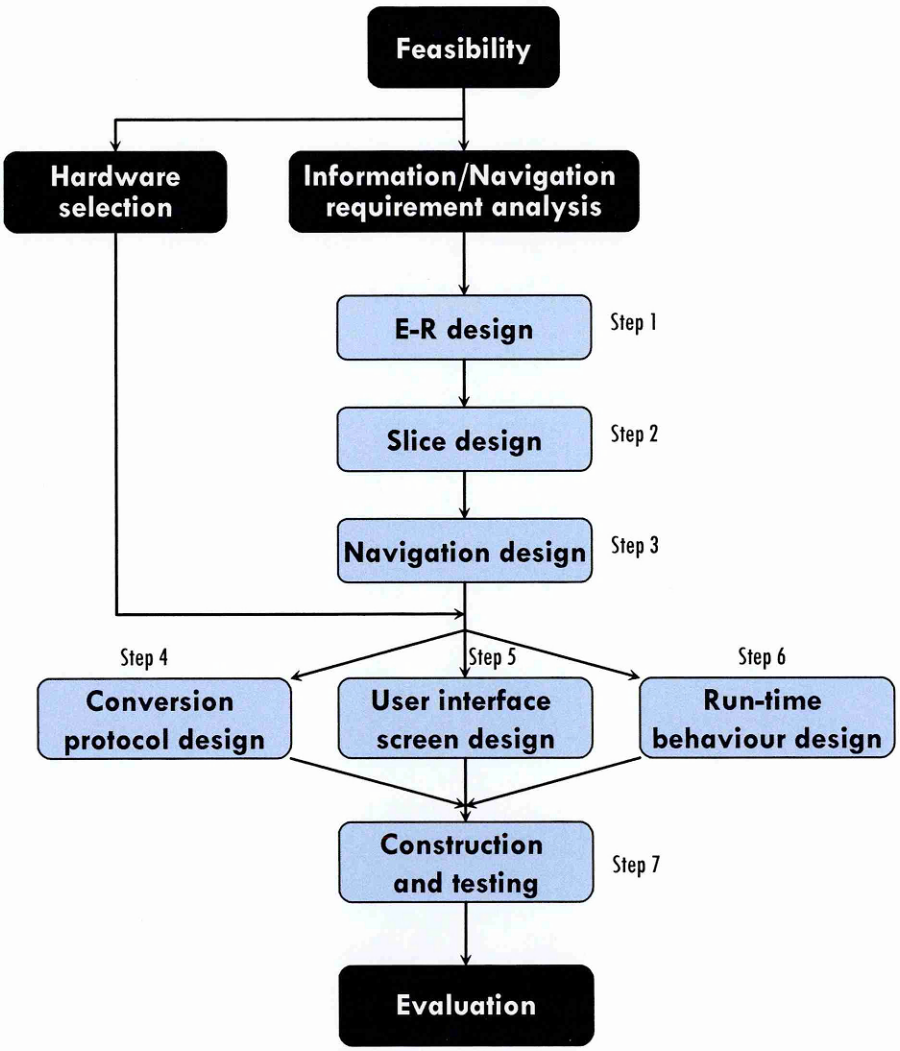


Figure 9 – Relationship Management Methodology [Isakowitz *et al*, 1995]

Conversion protocol design converts the entities and attributes of RMM into physical objects in a hypermedia application, for example table and fields. *User interface design* involves the design of the screen layout. *Run-time behaviour design* is the physical design of the program or the logic of the hypermedia application. *Construction and testing* include the selection of the hypermedia development environment. A Relationship Management CASE tool (RMCase) [Diaz and Isakowitz, 1995] was developed to support the development of hypermedia applications using RMM, although it is still not available commercially. However, if the CASE tool cannot be obtained, a conventional hypermedia authoring tool may be used in this step. After construction, the application should be tested to ensure that it fulfils functionality, navigation and usability requirements.

4.1.3 Object-Oriented Hypermedia Design Method (OOHDM)

The Object-Oriented Hypermedia Design Method (OOHDM) [Schwabe and Rossi, 1995] adopts an object-oriented modelling technique [Booch, 1994] and incorporates the HDM data model into design methodology. OOHDM allows the description of a complex information domain and specifies complex navigation as well as interface transformation.

A hypermedia application can be developed with OOHDM, using a combination of incremental, iterative and prototype-based design styles. Object-oriented models are developed in each step, improving on the models designed in previous iterations. The design method consists of four steps, as illustrated in Figure 10:

- ***Step 1: Domain analysis***. Uses the object-oriented technique to specify classes, subsystems, relationships and attributes, instead of using entity-relationships.
- ***Step 2: Navigation design***. Adds specific classes, e.g. node, link, and index, to represent different types of navigation.
- ***Step 3: Abstract interface design***. Similar to step 2, but is concerned with the user interface objects, e.g. buttons, text fields, etc.
- ***Step 4: Implementation***. The design of the actual application, including coding with programming languages.

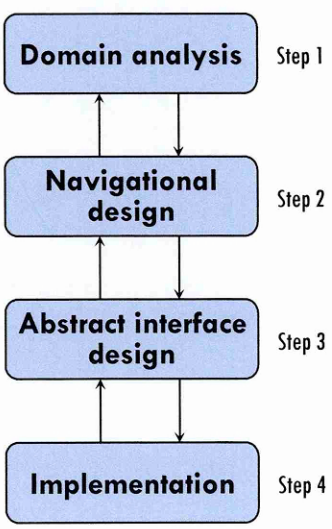


Figure 10 – Object-Oriented Hypermedia Design Methodology (OOHDM) [Schwabe and Rossi, 1995]

Domain analysis aims to analyse the overall application structure, using the object-oriented model. It includes the design of conceptual classes, object instances, abstractions and sub-classes. The output is domain classes, sub-classes and their relationships.

In *navigational design*, the designers develop all of the navigation structure of the application that reflects the user requirements and tasks. It uses navigational context and link notations to perform structured navigation of the domain classes from the previous step. Access to these navigational elements is modelled, using access structures such as indices and guided tours.

In *abstract interface design*, a model called *abstract interface* is produced, using text and multimedia objects, such as images, sounds and animations. By integrating navigational objects with interface objects, designers can try various interface models on the same navigational objects, for usability trials. OOHDM uses Abstract Data Views (ADV) charts [Cowan and Lucena, 1995] to model the static aspects of the user interface, while the dynamic aspects are modelled using techniques based on state diagrams.

Although both navigational and interface models can be implemented with existing hypermedia development tools, such as Macromedia Director, OOHDM-Web is the one recommended by Schwabe *et al* [1999].

4.1.4 Web Modelling Language (WebML)

The Web Modeling Language (WebML) is a visual notation that provides graphical and formal specifications, embodied in a design process for specifying hypermedia (or web) applications at the conceptual level [Ceri *et al*, 2002].

In WebML, the specification of a web application covers three main perspectives:

- Data model, which is the information assets managed by the application, their properties and logical correlations,
- Hypertext model, which is the hypertext interfaces of the application, consisting of pages for publishing content and of links for navigating and performing actions, and
- Presentation model, which is the graphical rendition of the interface.

The WebML *data model* is an adaptation of conceptual models for data design, as already in use in other disciplines, such as database design and software engineering. It is compatible with the Entity-Relationship data model [Chen, 1976] used in conceptual

database design and with Unified Modelling Language (UML) class diagrams [Booch *et al*, 1998] used in object-oriented modelling.

Hypertext model specifies composition and navigation of the site. *Composition* describes which pages compose the hypertext, and which content units make up a page. The web pages are the containers of information actually delivered to the user.

Presentation is the orthogonal task of defining the look and feel of pages in a site view. WebML does not include a specific model for expressing presentation at the conceptual level, but leverages standard approaches, more familiar to graphic and communication experts. Since WebML specifications can be represented using XML, presentation is considered like a document transformation mapping the WebML specification of a page into a page written in Java Server Pages (JSP) or ASP.NET [Ceri *et al*, 2003].

As shown in Figure 11, the WebML approach to the design of hypermedia applications consists of different phases that must be applied in an iterative and incremental manner. Nevertheless, the core of WebML design methodology is the *data design* and *hypertext design*. The process undergoes several cycles. Each cycle produces a prototype or a partial version of the application that allows conducting testing and evaluation as from an early phase.

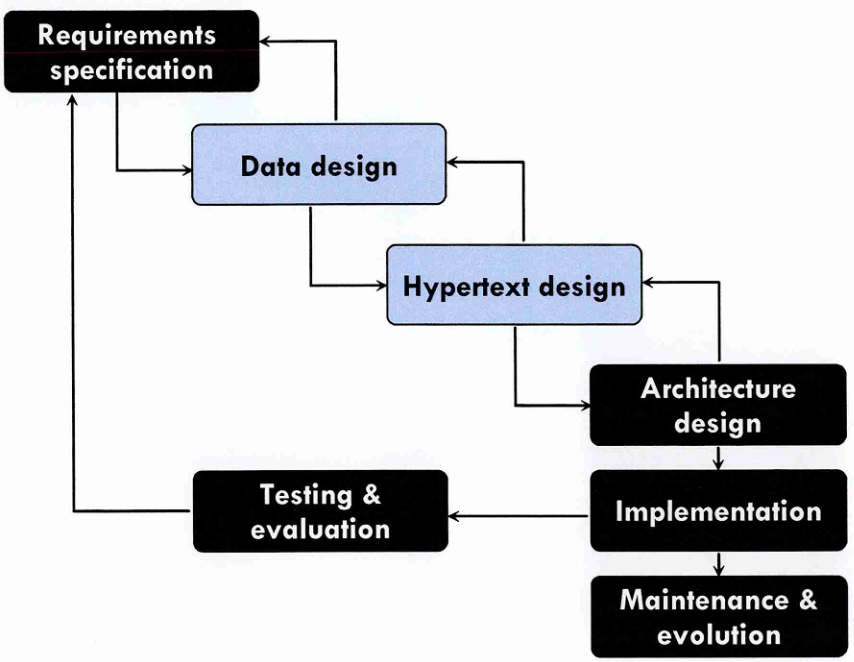


Figure 11 – WebML design methodology [Ceri *et al*, 2003]

4.1.5 Other design methodologies

So far, there are two types of hypermedia applications: *content-oriented* and *process-oriented* [Lang, 2001; Suh and Lee, 2001]. These categories can also be used to classify hypermedia development methodologies. Process-oriented methodologies are usually intended to assist the development of organisational processes, such as workflow management, whereas content-oriented methodologies concentrate on organising information content and the presentation of the hypermedia application to satisfy the user's cognitive processes.

Most of the well-developed techniques in hypermedia modelling and methodology, including RMM, OOHDM and View-based Hypermedia Design Methodology (VHDM) [Lee *et al*, 1999(a)], fall into the content-oriented category. VHDM considers that view support is highly desirable in designing hypermedia applications. VHDM is derived from RMM and presents 5 steps for designing hypermedia applications, based on the users' view. These steps are: requirement analysis, data modelling, view design, navigational design and mapping. Similar to RMM, *views* are generated from the entity-relationship diagram, and are used for subsequent navigational and mapping design. This methodology is especially effective for integrating enterprise databases with distributed hypermedia systems on the Internet.

Other methodologies, such as the Enhanced Object-Relationship Model (EORM) [Lange, 1994] and the Scenario-based Object Oriented Design Methodology (SOHDM) [Lee *et al*, 1999(b)], are categorised as process-oriented methodologies. EORM proposes the construction of a prototype of a user interface at an early stage during design. The methodology is based on three frameworks: class, composition and GUI (Graphical User Interface). The class framework consists of a reusable library of class definitions, identified by standard object-oriented techniques. The composition framework consists of a reusable library of link class definitions that enable users to reuse link classes and extend them when necessary by using inheritance. The GUI design determines the user interface aspect, for example the window in which the presentation has to appear. SOHDM has similarities with RMM, OOHDM and EORM, although SOHDM uses *scenarios*, instead of entities, and *scenario activity charts*, *events*, *activities* and *activity flows* primitives.

4.2 Hypermedia development methodologies

A full life cycle methodology usually involves a broader phase beyond the design phase that is the focus of the aforementioned methodologies. In addition, it also includes project management and maintenance of the hypermedia application. Therefore, the above methodologies are considered to be design methodologies, rather than development methodologies, which are commonly used in software engineering. The following sections will discuss in more detail the development methodologies for hypermedia applications.

4.2.1 Systems Development Life Cycle (SDLC)

Powell *et al* [1998] adopts the Systems Development Life Cycle (SDLC) method (also known as the *waterfall* model) from Royce [1970] to formulate the life cycle methodology for the development of a hypermedia application. The waterfall model is a sequential development phase that has distinct goals for each phase of development. The development moves from concept, design, implementation, testing, installation and troubleshooting, ending with operation and maintenance. The advantage of waterfall development is that it is more structured, so it allows easier control of the progress. This is because the developers have to plan everything up front.

However, this structured approach is also a weakness, since hypermedia applications developers are often uncertain as to what is required and how it can be implemented. Also, pure waterfall model does not allow for much revision. Once an application is in the testing stage, it is very difficult to go back and change something that was not well thought out in the concept stage. Therefore, a modified version of the pure waterfall model, where some steps can be carried out concurrently and by slightly overlapping the steps, is often preferable. In addition, the problem definition and requirement analysis are iterated like a whirlpool and then the rest of the steps can also be iterated when necessary. Therefore, the modified waterfall model is also known as the *whirlpool-modified waterfall* model [Powell *et al*, 1998].

The *spiral model* [Boehm, 1988] improves flexibility of the waterfall model by incrementally developing the application and breaking the development into many sub-projects. The purpose is to reduce risk by focusing on the most critical aspect of the application first and, with the knowledge gained from this step, to go back and implement the next most important features in small sub-projects. The idea is to turn several times, starting at the centre of the spiral first and expanding the step on each loop.

4.2.2 Hypermedia engineering approach

Lowe and Hall [1999] suggested a similar approach to the development of hypermedia applications considered from three different levels: process, method and technique.

A development *process* is a typical project management activity that needs to be carried out by a project manager. It involves the *analysis of problem domain*, *selection of product and process models*, *project planning* and, finally, the actual *development*. Product modelling involves the selection of a model for presenting information, the structure of information, application behaviour and functionality. The three different product models supported by the framework are the programming-based, screen-based and information-centred models. Process modelling consists of selecting phases, activities and artefacts for development and integration into the specific development process. Incremental development and prototyping are recommended for process modelling. As in any project management, the development process would likely be the responsibility of a project manager.

A development *method* encompasses the activities that are typically carried out by a systems analyst. It includes *systems analysis*, *design*, *production*, *verification* and *maintenance*. The focus of the systems analysis is to develop understanding and to record the results in a clear and understandable form, typically a specification document. Activities, such as requirement, user and task analysis, are very common in systems analysis. The design phase of the specification document is converted into a description on how to create an application that will fulfil the specification. The activities often include the content design, the design of the overall look and feel of the system, and development of a document for deployment. The components required for the hypermedia application will be produced during the production phase. The creation of pictures, forms, buttons and windows is typical of such activity. Verification is required to ensure that the application does what it is supposed to do. Testing will also need to be carried out to check functionality, performance, compatibility, reliability and, more importantly, usability. It is important to take into account maintenance, which may include corrective and adaptive maintenance.

Development *techniques* consider the practical aspects of how the development might be carried out. Although there are no prescribed methodologies, such guidelines are imperative to allow hypermedia developers or programmers to *formulate the requirement*, to *structure the information*, to *design the associative link* and *navigation support*, as well as to *design the user interface*. During the requirement analysis, the hypermedia developer may use information elicitation techniques, such as interviews, brainstorming, questionnaires and scenario analysis. Lowe and Hall [1999] suggested RMM [Isakowitz *et al*, 1995] as one of the most appropriate techniques to structure the information domain, particularly for applications that have a highly structured information model. Designing associative links is perhaps the most challenging aspect, because it relates to both usability and maintainability. Manual hyperlinks are acceptable for small-scale applications; however, automatic hyperlinks should be considered when the information domain is larger. Navigation support includes creating menus, indices and searching mechanisms. In designing user interface, the developers may experiment with different types of screen layout, frames, colours and fonts.

4.2.3 Industrial Hypermedia Development Methodology (IHDM)

Crowder *et al* [2000] proposed a methodology to develop industrial hypermedia applications that they believe to be flexible enough to accommodate current and any future manufacturing techniques and philosophies. They argue that current design methodologies, such as HDM, RMM and OOHDM, used alone, have many limitations in that they do not cover the early aspects of design process, nor do they address issues such as information reuse and many other activities during the development phase.

IHDM starts with requirement specification, followed by an iterative design process involving several stages: information structuring, authoring, navigation, user interface, access control and hardware selection. After the design stage, the process continues with prototyping, user trials and sign off design. The stages are illustrated in Figure 12.

The strength of IHDM lays in the authoring process, concentrating on information clustering to create modular hypermedia applications (MHA). This is helpful when developing a very large hypermedia application. A rule-of-thumb in software engineering suggests that a very large program can be more easily managed by decomposing it into smaller modules [Crowder *et al*, 1999].

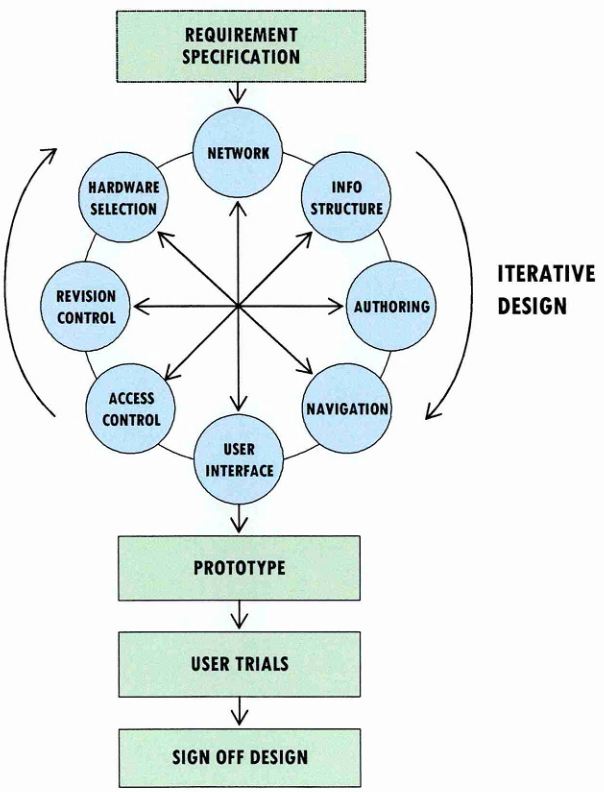


Figure 12 – Industrial Hypermedia Development Methodology [Crowder *et al*, 2000]

4.2.4 Hypermedia Flexible Process Modelling (HFPM)

Like Lowe and Hall's approach, Olsina, [1998] also proposes an engineering approach called Hypermedia Flexible Process Modelling (HFPM). It provides guidelines for planning and managing a particularly large hypermedia project. HFPM entails functional, methodological, informational, behavioural and organisational views of the hypermedia development process.

There are three general phases in HFPM. The first is the exploration phase, where initial concepts and user's needs are elicited. If necessary, a feasibility study is carried out in order to build the preliminary plan model. The second is the development phase. This phase is the core of the HFPM, consisting of:

- Requirement modelling, such as initial survey, use-case modelling and glossary construction
- Conceptual modelling, which is the analysis of the problem domain
- Navigational modelling, which comprises the analysis of intended users' tasks, identification of navigational classes and specification of navigational schema
- Abstract interface modelling that refers to the analysis of user interface models
- Physical modelling, comprising run-time component integration, prototyping and storyboarding
- Validation/verification, consisting of an experimental feedback loop between developer and users
- Quality assurance, such as analysis of improvement strategies
- Documentation

The third is the operational phase, which essentially consists of product configuration, maintenance and evolution of the hypermedia application. Figure 13 shows the stages in HFPM.

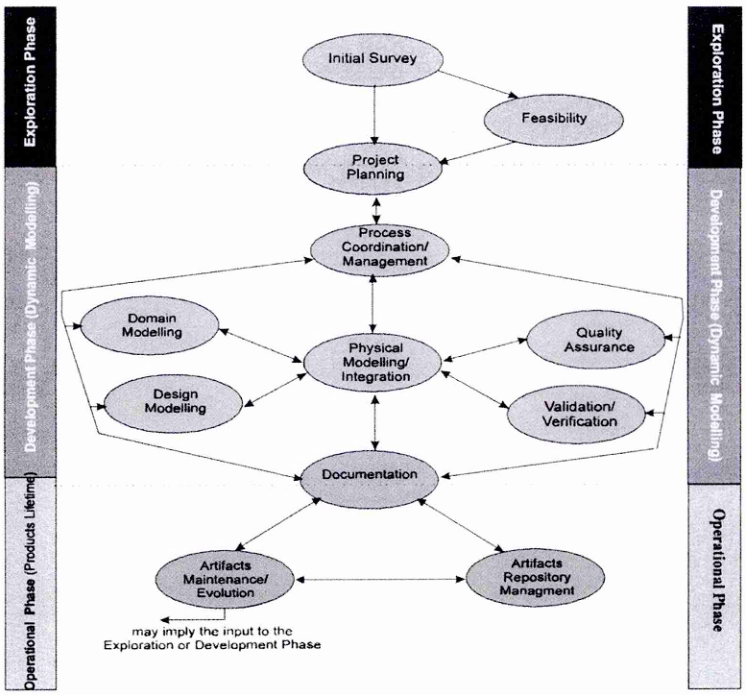


Figure 13 – Hypermedia Flexible Process Model [Olsina, 1998]

4.2.5 Rational Unified Process (RUP)

Within the context of software engineering, Rational Unified Process (RUP) [Jacobson *et al*, 1999] can denote twofold: a software development approach and a well-structured software engineering process [Kroll & Kruchten, 2003].

As a software development approach, RUP is use-case driven, iterative and architecture-centric. Like Boehm's *spiral model*, the main purpose of RUP is to mitigate risk of failure by iterating the requirement specification, analysis, design, implementation and testing (Figure 14). Each iteration builds on the work of previous iterations to produce a prototype that is one step closer to the final product. Early iterations focus on requirement specifications, analysis and design, while later iterations will have a more emphasis on implementation and testing.

As a software engineering process, RUP is modelled using the Software Process Engineering Metamodel (SPEM), a process modelling based on the UML. The architecture of RUP follows two axes as illustrated in Figure 15.

- The horizontal axis represents the dynamic aspect (or time dimension) of the process, as time passes, and it is expressed in terms of cycles, phases, iterations and milestones.

- The vertical axis represents static aspect of the process. It describes how processes elements, which include process components, activities, artefacts and roles, are grouped logically into core process disciplines (or workflow).

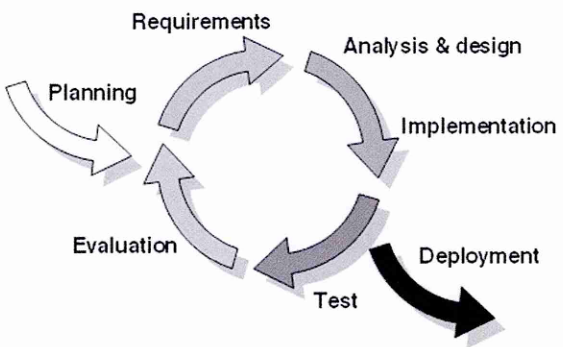


Figure 14 – Iterative development in RUP [Kroll & Kruchten, 2003]

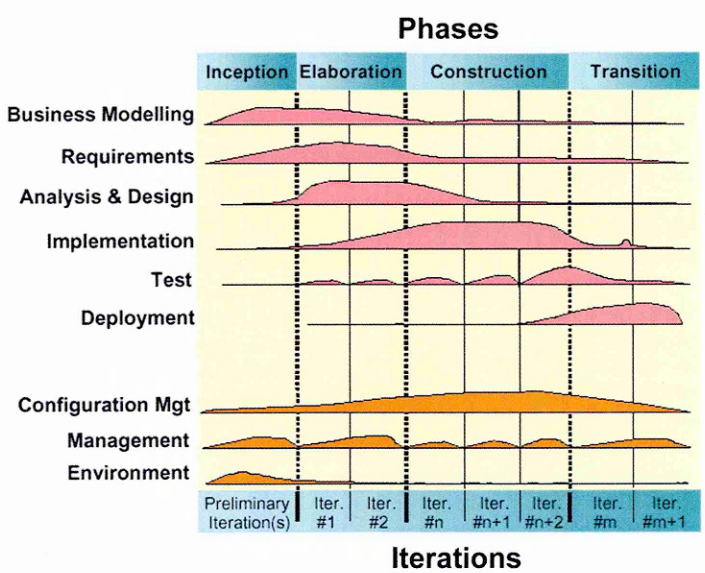


Figure 15 – Two dimensions of the RUP [Kroll & Kruchten, 2003]

The dynamic aspect of RUP deals with the lifecycle of a project that can be divided into four phases: *inception*, *elaboration*, *construction* and *transition*. Inception establishes an understanding of all the requirements and scope of the project. Elaboration includes

most of the technical tasks, such as design, implementation and testing. Construction does most of the implementation tasks from an executable architecture to the first operational versions (alpha releases). Transition ensures that software fulfils the requirements of the end users, including testing and preparation before release and making minor adjustments.

The static aspect of the RUP describes *who* is responsible for doing *what*, *how* things are done and *when* it should be done. These are also known as the key modelling elements in RUP.

In addition to the development framework and process, RUP is also a software tool. RUP is designed and documented using the UML. Figure 16 shows a screenshot of the RUP.

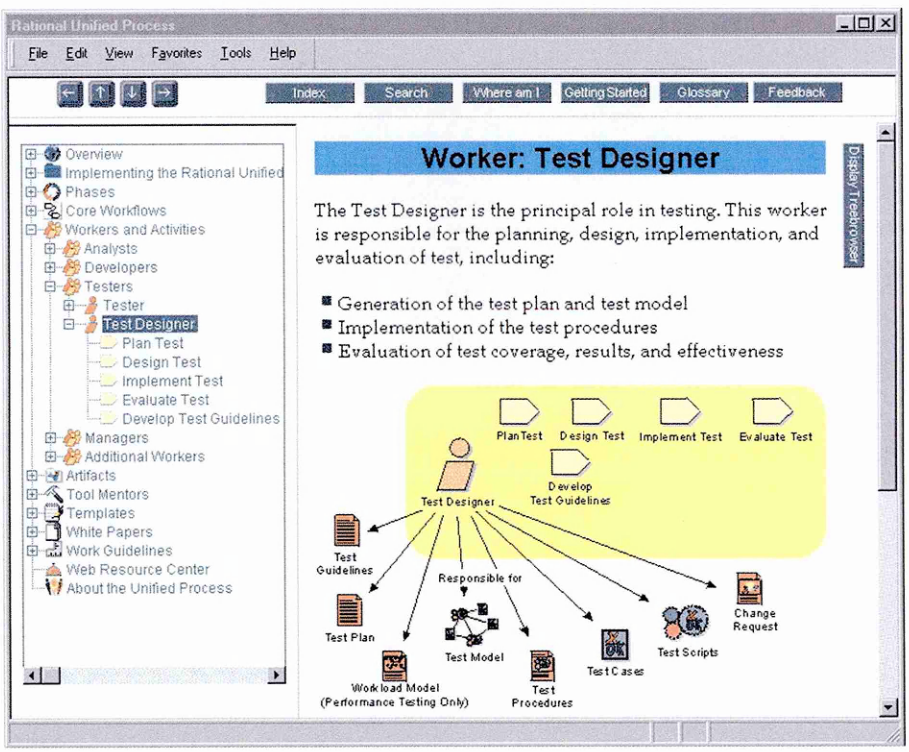


Figure 16 – Screenshot of the RUP

4.3 Comparison of the methodologies

So far, a survey of literature on hypermedia design methodologies has given two options for the formal design methodologies: one based on Entity-Relationships modelling and the other based on Object-Oriented modelling. The former leads to methodologies such as HDM, RMM and VHDM, while the latter is adopted by methodologies such as OOHDM, EORM and SOHDM.

Due to the fundamental differences in how these models originated and evolved, for example, EORM and SOHDM originated from OOHDM, only four design methodologies and five development methodologies will be compared in this section. They are HDM, OOHDM, RMM, WebML, SDLC, Lowe and Hall's engineering approach, IHDM, HFPM and RUP. The comparison will be based on the following criteria: process, modelling technique, graphical representation, notation, database support, tools support and support for user interface design.

Table 4 shows a comparison of the four design methodologies. All of them use formal notation and graphical representation. Both HDM and RMM are based on traditional yet well-known E-R modelling, whereas OOHDM uses a more contemporary object-oriented technique. As such, the notation used in OOHDM is UML, as opposed to E-R diagram in HDM and RMM. WebML methodology, on the other hand, uses proprietary diagram, which combines of E-R and UML. They all support database design to be used in the hypermedia application, although only OOHDM gives support for user interface design. Finally, although OOHDM, RMM and WebML have tools to assist the developer from the logical to the physical design of the hypermedia application, they are not available commercially[2].

In Table 5, five development methodologies are compared. All of the development methodologies have similar features in that they are all based on the SDLC and the spiral model, in which each step is iterated to minimise the risk of failure. The main difference between Lowe and Hall's methodology and the HFPM occurs in the design phase. Lowe and Hall [1999] suggest that RMM be used during analysis, design and implementation, whereas HFPM adopts OOHDM. Both SDLC and Lowe and Hall's methodologies prescribe design rationale for user interface, navigation and information structuring. HFPM, on the other hand, does not provide the design rationale, since it relies on OOHDM design steps.

Considering the development phases, HFPM is also known a broad hypermedia engineering approach. It covers all essential phases and activities of a hypermedia project and hence helps to establish milestones of the project, as well as promoting communications.

[2] During the research execution and thesis writing, OOHDM-Web, RMCase and WebRatio, which support OO-HDM, RMM and WebML respectively, were not available commercially

IHDM appears to be the most comprehensive methodology, but in Crowder *et al* [2000] prototyping and sign-off stages were not described in detail, only the ability to customise the user interface was discussed.

RUP is well-known in software engineering and has been adopted as industry standard for software development and project management. It is similar to Boehm's spiral model and uses the UML notations. RUP is well-suited in a project where many people are working together to develop a large-scale software application.

Table 4 – Comparison of hypermedia design methodologies

Methodology	*Process*	*Modelling technique*	*Graphical representation*	*Notation*	*Database design support*	*User interface design support*	*Tool support*
HDM [Garzotto *et al*, 1993]	1. Authoring in the large 2. Authoring in the small	E-R	1. & 2. E-R diagram	E-R	Not specified	Not specified	Not specified
OOHDM [Schwabe and Rossi, 1995]	1. Conceptual design 2. Navigation design 3. Abstract user interface design 4. Implementation	OO	1. Class diagram 2. Navigational class + context schema 3. ADV configuration diagram + ADV chart	1. UML 2. own 3. ADV	OO-database	ADV chart	OOHDM-Web
RMM [Isakowitz *et al*, 1995]	1. E-R design 2. Slice design 3. Navigational design 4. Conversion protocol design 5. User-interface design 6. Run-time design 7. Construction & testing	E-R	1. E-R diagram 2. Slice diagram 3. RMDM diagram	1. E-R 2. & 3. proprietary	RDBMS	Not specified	RMCase
WebML [Ceri *et al*, 2003]	1. Requirement specification 2. Data design 3. Hypertext design 4. Architecture design 5. Implementation 6. Testing & evaluation 7. Maintenance & evolution	E-R	2. & 3. WEbML diagram	2. & 3. WebML	DBMS & XML	WebML	WebRatio Site Development Studio

Table 5 – Comparison of hypermedia development methodologies

Methodology	*Process*	*Modelling technique*	*Graphical representation*	*Notation*	*Database design support*	*User interface design support*	*Tool support*
Semi-structured SDLC [Powell *et al*, 1998]	1. Requirement analysis 2. Design 3. Implementation 4. Testing 5. Maintenance	Not specified	Not specified	Not specified	Not specified	Yes [Light prototyping]	Generic hypermedia tools: HTML, Director
Engineering approach [Lowe & Hall, 1999]	1. Domain analysis 2. Product modelling 3. Process modelling 4. Project planning 5. Development -Analysis -Design -Requirement analysis -Info structuring -Linking mechanism -Access support -Integration -Production -Testing -Maintenance 6. Documentation	5. - RMM [suggested]	No specified	Not specified	Not specified	Partial [Design of the overall look and feel]	Generic hypermedia tools: HTML, Director

Table 5 – Comparison of hypermedia development methodologies [*continued*]

Methodology	*Process*	*Modelling technique*	*Graphical representation*	*Notation*	*Database design support*	*User interface design support*	*Tool support*
IHDM [Crowder *et al*, 2000]	1. Specification -Requirement specification -Feasibility study 2. Design (iterative process) -Authoring, info structure, navigation, user interface, access control, revision control, network, hardware 3. Prototyping 4. User trials 5. Sign off design	Not specified	Not specified	Not specified	Yes [using Microcosm]	SHEP [Hall, 1995]	Microcosm
HFPM [Olsina, 1998]	1. Exploration phase -Requirement elicitation -Feasibility study 2. Development -Requirement modelling -Conceptual modelling -Navigational modelling -Abstract interface modelling -Physical modelling -Validation/verification -Quality assurance -Documentation 3. Operational phase -Product configuration -Maintenance	OO	2. - OOHDM	OOHDM	Not specified	Not specified	Not specified

Table 5 – Comparison of hypermedia development methodologies [*continued*]

Methodology	*Process*	*Modelling technique*	*Graphical representation*	*Notation*	*Database design support*	*User interface design support*	*Tool support*
RUP [Jacobson *et al*, 1999]	1. Inception -Requirement specification -Scope of project 2. Elaboration -Analysis, design, implementation and testing 3. Construction -implementation tasks 4. Transition -Testing & preparation before release	UML	Use-case, class diagram, state diagram, activity diagram	UML	Not specified	Not specified	RUP

4.4 Design rationale for hypermedia TSS

Thüring *et al* [1995] point out that, in general, hypermedia applications serve two fundamental purposes: (1) as 'hyperbases' and (2) as 'hyperdocuments'. A hyperbase application is also known as an information retrieval tool, which usually offers the user unconstrained search capability. In contrast, a hyperdocument incorporates various documents that intentionally guide the user through the information space, along a predefined structure. The latter is appropriate for tasks requiring deep understanding and learning. Therefore, the developers of hypermedia task support systems should pay attention to the design rationale that directly affects the user's comprehension.

According to Thüring *et al* [1995], comprehension is usually characterised by the user's ability to construct a mental model that represents relationships between information objects. Therefore, comprehension may be improved by two design principles: increased *document coherence* and reduced *cognitive overhead* (Figure 17). The former can facilitate the construction of semantic relationships between information objects, whereas the latter can release the user from disorientation, usually caused by poor user interface and navigation design.

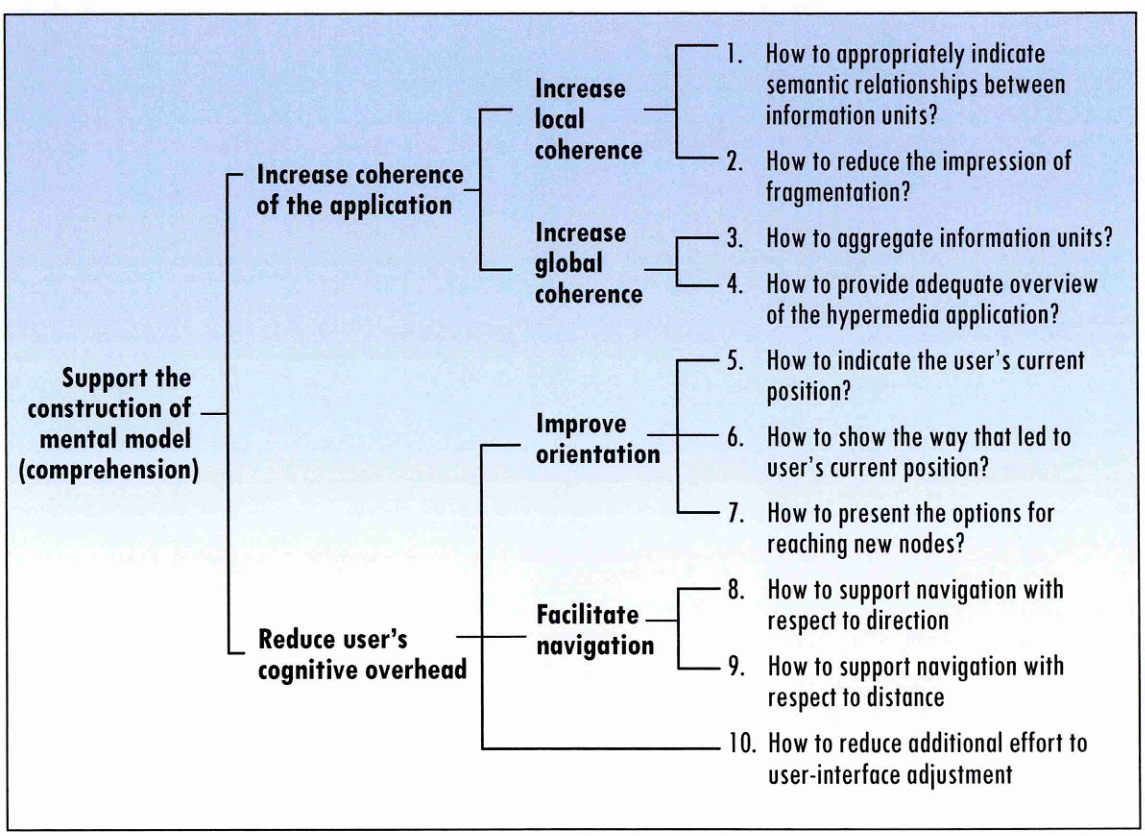


Figure 17 – Hypermedia design issues [Thüring *et al*, 1995]

Local coherence means that users deduce small-scale connections that link sentences and clauses into meaningful information, whereas global coherence is generated through a conclusion drawn from several sentences, paragraphs, or even chapters [Thüring *et al*, 1995]. The terms orientation and navigation imply that the hypermedia application contains spaces where users move from one piece of information to another. If the application is complex, it can lead to disorientation, known as 'being lost in hyperspace' [Nielsen, 1995], where users do not know where they are, or where to go next.

To support the comprehension of hypermedia applications, Thüring *et al* [1995] propose eight design principles that can enhance formal design methodologies, such as RMM and OOHDM, and which may also be used as guidelines for evaluation. The eight design principles, and the issues addressed, are given in Table 6.

Table 6 – Design principles and their relation to cognitive issues [Thüring *et al*, 1995]

	Principles	*Issues addressed*
P1	Use typed link labels	1*, 2
P2	Indicate equivalencies between information units	1, 2*
P3	Preserve the context of information units	1, 2*
P4	Use higher order information units	3
P5	Visualise the structure of the document	1, 2, 3, 4*, 8, 9
P6	Include cues into the visualisation of structure, which show the user's current position and navigational options for moving on	5, 6, 7
P7	Provide a set of complementary navigation facilities, which cover aspects of direction and distance	8, 9
P8	Use a stable screen layout with windows of fixed position and default size	10

* indicates the main concern

Experimental studies and observations on how people actually conduct the design process have suggested that the final design is the result of a long and recursive process, involving thinking about ideas, production, reorganisation, modification and evaluation [Nanard and Nanard, 1995]. Based on this argument, the developer should also consider *cognitive* and *mental* activities during the development process rather than solely following the formal methodology. In fact, these two aspects are complementary and both are equally important. Furthermore, they suggest that the design or development process should be illustrated on two axes: one axis represents formal methodology and the other one represents mental process (Figure 18).

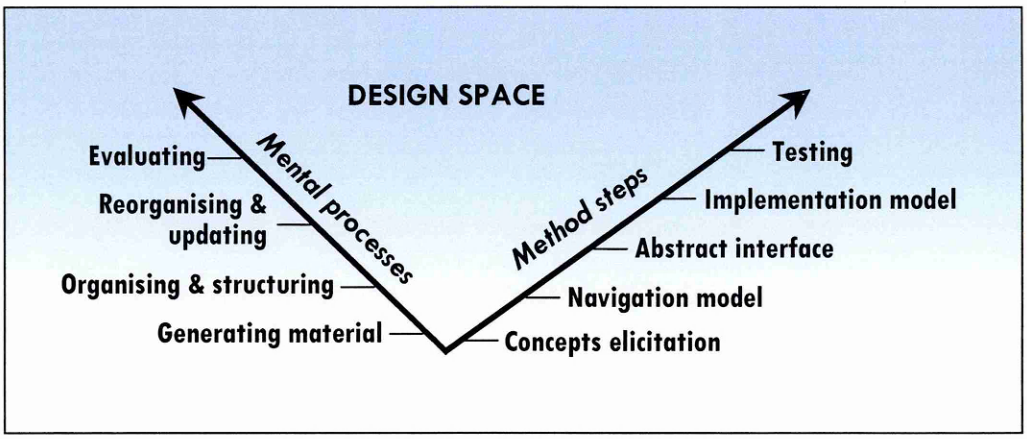


Figure 18 – The hypermedia design activity [Nanard and Nanard, 1995]

The mental processes include generating materials, structuring, updating and evaluating the hypermedia application. The method steps are similar to steps in other hypermedia design methodologies, such as RMM and OOHDM. They include concept elicitation, navigation model, abstract interface, implementation and testing. Nanard and Nanard [1995] suggest that, during the design process, the hypermedia designer should use both axes in Figure 18 by switching between the activities whenever necessary.

Furthermore, in developing a hypermedia application, the concept of design space implies that the formal techniques or methodology to support abstract model and conceptualisation must support the designer's incremental and recursive activities. Therefore, the hypermedia development environment should support:

- ***A fast experimental feedback loop*** to assist in the evaluation process and design recursion.
- ***Tools*** that enable the developers to swap from a conceptual model to a physical model, thus promoting both structuring and updating activities.
- ***Cloning tools,*** at instance or physical level, that enable the generation and duplication of material activity
- ***Mechanism*** that allows evaluation to be carried out at any stage of the formal method
- ***Light prototyping*** to produce the basic layout and navigation within the hypermedia application at any stage of the development.

Although, in general, hypermedia applications can be modelled using software engineering techniques and their derivations, the most important part of the hypermedia

design, namely aesthetic and cognitive issues, are not supported by the software engineering environment [Nanard and Nanard, 1995; Lowe and Hall, 1999]. *Experimental feedback* is therefore very crucial because, unlike the ordinary software user, the hypermedia user is generally broader based and it is necessary to check that the application meets this requirement. On the other hand, *formal tools* are also required to ensure the thoroughness and consistency of the application and could help in reducing updating and maintenance effort. In practice, the needs for formal tools and for experimental feedback are not equal at each stage of the development process. Instead, both have their own focus. Formal tools are required during the early stage of the development process, whereas experimental feedback would be more important at the application stage.

4.5 Hypermedia success model

A success model provides a reference against which the criteria will be compared and therefore is required before conducting evaluations of hypermedia applications. A number of existing success models will be reviewed from the literature on Management Information Systems (MIS), Human-Computer Interaction (HCI) and hypermedia.

4.5.1 DeLone and McLean model

Developing the Information Systems (IS) success model has been a major research topic in the area of Management Information Systems (MIS), at least since the early 1980s. There were, therefore, a large number of measures used to evaluate success. Perhaps the most significant contribution to the MIS community was the work of DeLone and McLean [1992]. They reviewed about a hundred articles and summarised the various success measures into six dimensions of information system success. They are: system quality, information quality, use, user satisfaction, individual impact and organisational impact. Based on these, DeLone and McLean [1992] suggested an information system success model shown in Figure 19.

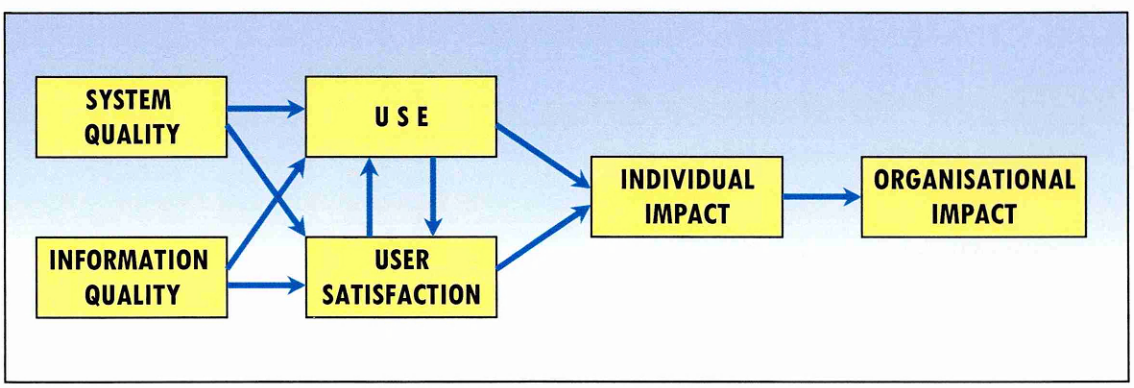


Figure 19 – IS success model [DeLone and McLean, 1992]

The model asserts that the information system will only be successful if it has a positive impact on the organisation. In order to do this, the system should first impact on the individual as the main user, as a direct effect of both the amount of the system's use and of user satisfaction. The system's use and user satisfaction are believed to have a multidirectional impact, in that the amount of use can affect user satisfaction and, equally, user satisfaction influences the amount of use. Finally, both use and user satisfaction are the result of system quality and information quality.

After the publication of DeLone and McLean's paper, the proposition of the success model has become even more popular within the MIS community, resulting in the adoption of use and/or user satisfaction as the determinants of success. However, if the effect of user satisfaction on various types of information systems is to be measured, system quality and information quality must also be measured, as they vary from system to system.

4.5.2 Lucas model

Similar to DeLone and McLean [1992], [Lucas, 1978] also claims that the extent to which the system is *used* and the *user's satisfaction* are the two most common factors determining the successful implementation of an information system. He conducted studies on information systems implementation and discovered that there are six factors that also influenced the system's use:

1. *User's personal stake* relates to how relevant the system is to the work that the user has to carry out. *User's personal stake* itself is influenced by the level of support from management and by the urgency of the problem that needs to be addressed. The higher the level of management support and the more urgent the problem, the higher the user's personal stake in the system.

2. *System characteristic* includes such aspects of the system's design as ease of use, reliability and relevance to the task that the system supports

3. *User demographic* refers to the characteristics of the user, such as age and degree of computer experience.

4. *Organisation support* is the degree of support provided by management, for example through a better infrastructure and working environment. The better the support infrastructure, the more likely an individual will be to use the system.

5. *Performance* is the extent to which the individuals can do their task with the support from the system. The more users can do with a system and the more benefits the user can get from that system, the more they will use it.

6. *Satisfaction* has a two-way relationship with use. The more satisfied the user, the more they will use the system. The more they use the system, the more satisfied they are.

The factors identified by Lucas [1978] and the relationships they have to each other are illustrated in Figure 20. Note that the arrows showing the relationships between use, performance and satisfaction have two directions, indicating the two-way relationships between these factors.

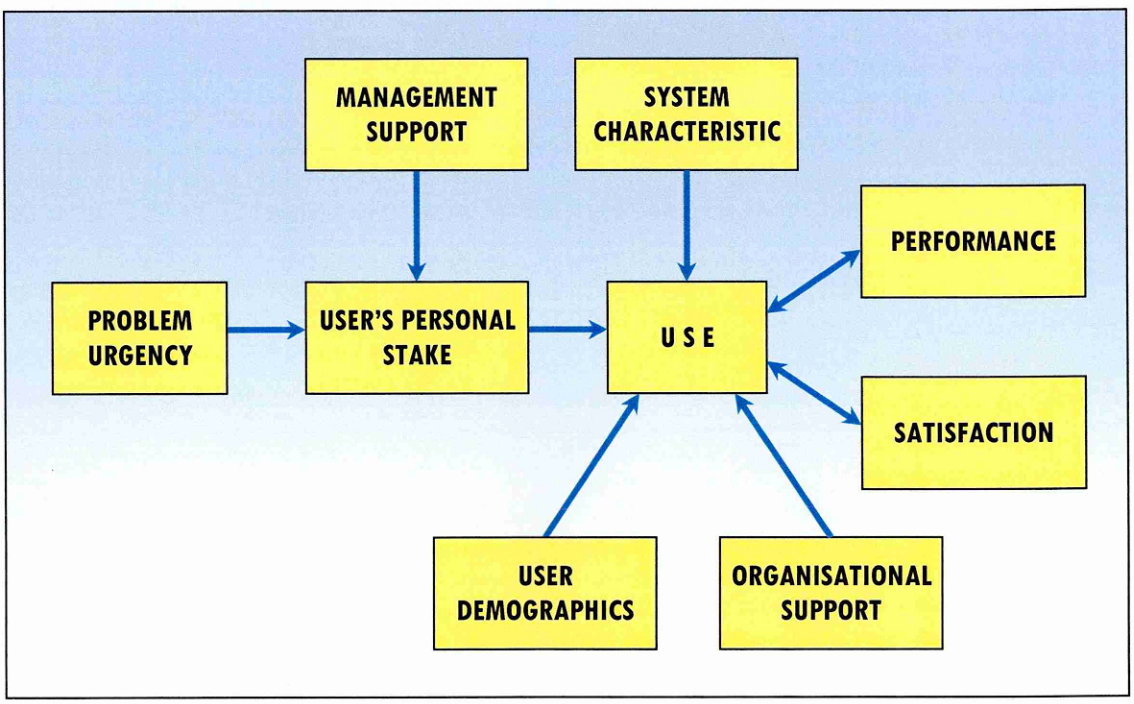


Figure 20 – Implementation success model [Lucas, 1978]

4.5.3 Technology Acceptance Model (TAM)

The Technology Acceptance Model (TAM) [Davis, 1989] was devised specifically to model *user acceptance* of an information system and to provide an explanation of the determinants of computer acceptance in general. TAM (shown in Figure 21) postulates that the use of an information system is fundamentally determined by *behavioural intentions* (BI) to use the system. Furthermore, BI is jointly determined by two other constructs: *perceived ease of use* (EOU) and *perceived usefulness* (U).

TAM consists of three major constructs:

- *Perceived ease of use* (EOU) is the degree to which the prospective users expect the system to be free from effort, which may include learning, physical and mental effort

- *Perceived usefulness* (U) is the prospective users' subjective expectation that using a specific system will increase their job performance. It comprises three elements: effectiveness, productivity and efficiency (time saving).

- *Behavioural intention to use* (BI) is the prospective users' decision to accept or to reject the system, indicated by their intention to use that system. BI is the variable that depends on EOU and U

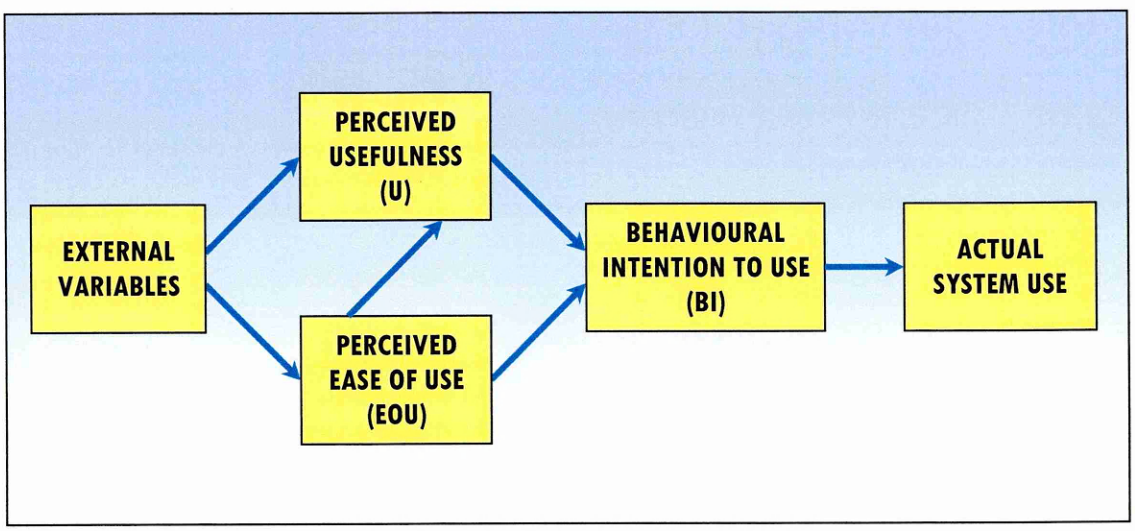


Figure 21 – Technology Acceptance Model [Davis, 1989]

Davis *et al* [1989] empirically tested the TAM, using a number of MBA students as the subjects, operating a word processing package. The constructs in TAM were measured using a multiple-items instrument (questionnaire). The findings of the study supported Davis' [1989] model, claiming that the model was a powerful tool to predict and to explain the users' behaviour towards using the information system.

Following this study, TAM became one of the most popular models to be used in various information system evaluations. Fakun [2000], in his research, summarised 25 studies undertaken using TAM. He also extensively employed TAM to evaluate industrial hypermedia applications, and confirmed from his review that the TAM's constructs are reliable and valid. Fakun [2000], quoting Neuman [1997], describes reliability as the degree to which an instrument is consistent when measuring a construct, whereas validity determines whether an instrument is actually measuring what it is intended to measure.

4.5.4 Task-Technology Fit (TTF) model

The Task-Technology Fit (TTF) model was originated by Goodhue [1995]. He defined TTF as the degree of fit to which technology assists an individual in performing their portfolio of tasks, i.e. correspondence between task requirements, individual abilities and the functionality of the technology. TTF also implies the matching of the functionality or capability of the technology (or IT system) to the demands. TTF postulates that the IT system will be used if, and only if, the functions available to the user support the activities of the user [Goodhue, 1995]. Consequently, the system that does not offer sufficient advantage will not be used.

The TTF model (shown in Figure 22) provides a theoretical foundation for investigating factors that affect the degree of fit between *task characteristics* and *technology characteristics* or IT system functionalities. With these factors, the study can investigate the impact of low or high degree of fit on system utilisation, and which factors influence the fit between task and hypermedia system characteristics. Another advantage of the TTF construct is that the model captures objective beliefs that the tool or technology will assist users to carry out their task. This means that, although a person does not like the software, that person may still use it, as it will lead to the completion of the task [Dishaw and Strong, 1998].

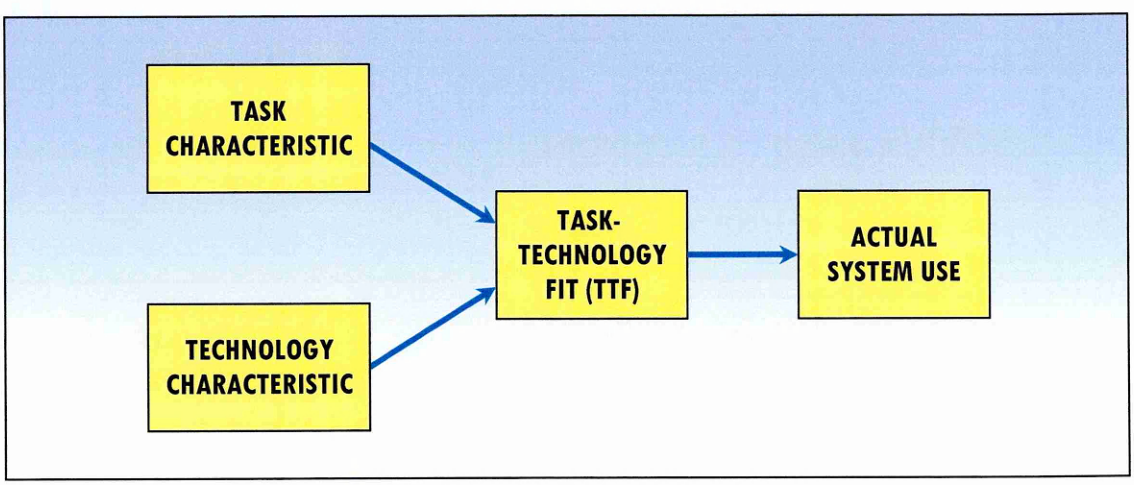


Figure 22 – Task-Technology Fit (TTF) model [Goodhue, 1995]

The TTF instrument devised by Goodhue [1998] has 12 dimensions (Table 7). He tested the reliability and validity of the instrument on an organisation's information system, one that supported decision making for its managers. The 12 dimensions proved to be reliable and exhibited discriminant and predictive validity [Goodhue, 1998]. Reliability implies that the dimensions are dependable and consistent. Discriminant validity means that responses to one construct are not associated with another, different, construct, implying that each dimension can be used on its own. Predictive validity is when an indicator predicts future events that are logically related, i.e. the degree of TTF predicts the level of utilisation.

Table 7 – The 12 dimensions of Task-Technology Fit (TTF) [Goodhue, 1998]

Dimension	*Description*
Lack of confusion	Ease of knowing how to use the information
Level of detail	Information is maintained at the right level of detail
Meaning	Ease of determining what the information means
Locatability	Ease of determining where to get the required information
Accessibility	Easy access to desired information
Assistance	Ease of getting help on problems with information
Ease of use	Ease of working with the system
System reliability	Dependability of access and up time of systems
Accuracy	Correctness of the information
Compatibility	Information from different sources can be compared without inconsistencies
Currency	The most up-to-date information
Presentation	Information is presented in readable and useful format

The TTF model has also been used by a number of researchers to investigate Information System usage. Dishaw and Strong [1998], for instance, employed the TTF model to assess the utilisation of a software maintenance tool. The subjects of the study were professional computer programmers, working in aerospace, insurance and financial services companies. The main tasks for those companies were similar, i.e. supporting maintenance of both software and data. The results emphasised the importance of understanding fit, so that the software developer can draw the aspects of fit that lead to system utilisation.

Mathieson and Keil [1998] asserted that the degree of fit between the task and the system might affect the perceived ease of use. They used a laboratory experiment involving undergraduate business students enrolled on an IT course. The experimental task was to create database queries in order to answer questions in a given problem domain, using two different systems. The results implied that the perceived ease of use is another factor, besides TTF, which affects system utilisation.

A study on a Group Support System (GSS) showed how a poor fit of the GSS with the group's task affects the group's performance [Zigurs *et al*, 1999]. They found that the GSS groups typically perform better than non-GSS groups, when the GSS profile and the task is a match.

4.5.5 Hypermedia usability

The Hypermedia success model has been largely derived from the assessment of its usability from the Human-Computer Interaction (HCI) perspective. Although there has been a relatively large amount of research work focusing on the evaluation of interactive hypermedia applications, probably the most notable has been the work by Nielsen [1995]. He employs a system acceptability model, shown in Figure 23, to deduce the hypermedia success model.

Nielsen [1995] asserted that the acceptance of the system is a combination of its social acceptability and its practical acceptability. The former implies the extent to which the hypermedia application fits with the current social environment, whereas the latter is associated with functional criteria such as cost, reliability, compatibility and usefulness.

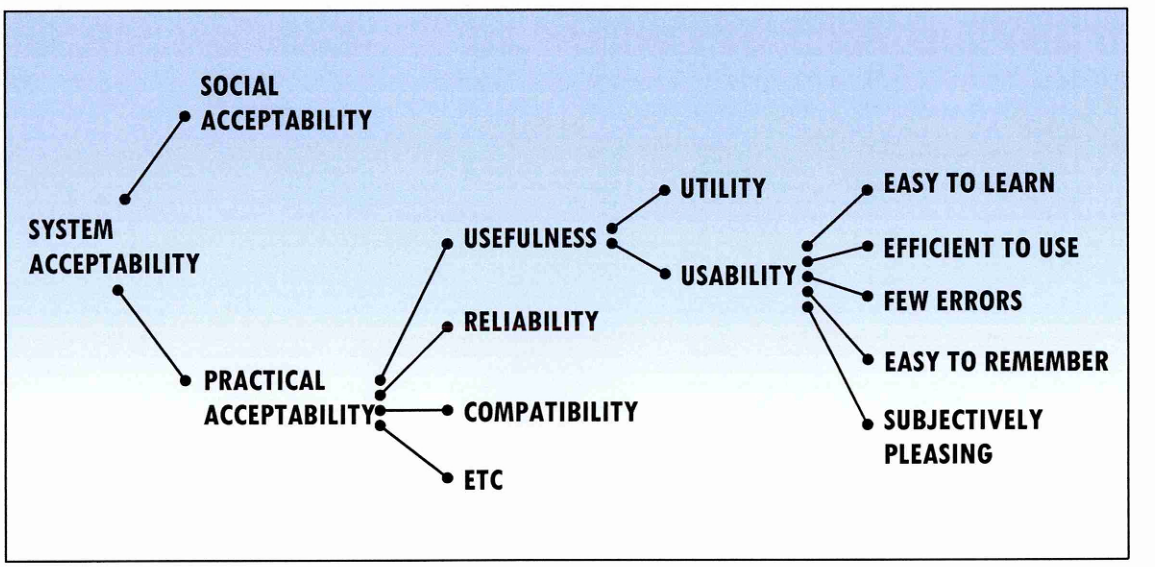


Figure 23 – System acceptability model [Nielsen, 1995]

Usefulness is defined as the matter of whether or not the system can be used to achieve some desired goal. Nielsen [1995] further divided usefulness into utility (whether the functionality of the system can do what is required) and usability (how well the users can use that functionality).

Usability for hypermedia applications depends on the usability of the underlying hypermedia system, the usability of the content of the information, and how well those two elements fit together. Nielsen [1995] suggests that the fit is assessed using the parameters: easy to learn, efficient to use, easy to remember, few errors and pleasant to use. These parameters are described in more detail in Table 8.

Table 8 – Attributes of hypermedia usability [Nielsen, 1995]

Attributes	*For end users*	*For authors**
Easy to learn	How quickly the first time users understand the first screen they see and can look for specific information	How easily the authors can understand the basic structure of the hypermedia system and are able to modify it without knowing the entire content of the application
Efficient to use	How quickly the users can find the information, or realise that the information may not be found in a particular location	How quickly the authors can construct a hypermedia structure to reflect their understanding
Easy to remember	How easily the users can remember how to navigate and to get the same information after a period of not having used the hypermedia application	How easily the authors can return to the information base and update it, if the hypermedia structure needs revision after some time
Few errors	How easily the users can return to their previous location, should they accidentally follow a link	The number of errors that a hypermedia application may have, that lead to a location where the end users are not supposed to go. How accurate the information attached to a hypermedia node.
Pleasant to use	Users feel that they are in control of the hypermedia system and as such they prefer to use the application to existing alternative solutions, such as paper or other non-hypermedia systems	Authors feel in control of the authoring environment ensuring they are not constrained by the limitation of the system

*who provide content of the hypermedia application

4.6 Testing and evaluation methodology

Testing is often associated with finding errors and, ultimately, ensuring that the application functions properly. Depending on the size and the complexity of the application, the time when testing should be conducted may vary. It can start as soon as coding started, or can begin during or after implementation. Nevertheless, developing a test plan is crucial to understanding what needs to be tested and how to collect the test data.

Mosley [1993] categorises the types of testing according to whether they employ static or dynamic techniques, and whether the test is manual or automated. Static testing means that the coding being tested is not executed, whereas dynamic testing involves execution of the code. Manual testing requires a human tester to carry out the test, while in automated testing, a computer conducts the test.

In static testing, inspections are conducted formally, by participants, to manually examine code for the occurrence of well-known errors. Similarly, syntax checking attempts to find syntax errors and some other routine errors, but it is conducted automatically under the control of computer and programming language. In dynamic, manual testing, both walkthrough and desk checking aim to find errors in the code by verifying the logic and following the logic step-by-step.

The first technique in dynamic and automated testing is unit test, sometimes called a module test. It aims to find errors that may exist in a module by testing each module alone. Since modules usually work together, they must be tested together in a larger group at a later stage. Combining modules and testing them is called an integration test. In a system test, all elements of the application are tested and integrated into a system.

The word 'evaluation', rather than 'testing', is often used to describe the 'soft' approach to testing user acceptance of an application. In the context of the System Development Life Cycle (SDLC), evaluation is an iterative process, which provides the developer with a clear understanding of what the users require and whether or not the system has fulfilled the users' requirements. In this respect, evaluation provides a feedback mechanism that allows control of the development cycle. In order to carry out an effective evaluation, different evaluation techniques are required during the various stages of the development, depending on the objectives of the evaluation.

In general, two types of evaluations are often conducted: *formative* and *summative* [Faulkner, 2000]. Formative evaluation is commonly used during the design process, to gather ideas from the prospective users and to check and re-check the design. This implies that the developer is required to work closely with the prospective users, to effectively elicit their opinions about various aspects of the application. During the formative evaluation, the application itself is often in the form of a prototype or rough design, and therefore it will focus on gathering qualitative aspects of the application.

Questions asked of the users may include: how they feel about the application, what features that they like or dislike most, what problems they experienced, ideas for further improvement, etc.

Summative evaluation, on the other hand, is usually conducted when the application is complete or when the major building blocks and functionalities of the application can be demonstrated to the prospective users in a presentable manner. The objective of this type of evaluation would be to assess the overall functionality of the application and its usability performance, using quantitative measures. Therefore, a quantitative technique, for instance a questionnaire or a survey, is often used to collect the data to provide the actual measures of what improvements have been made to the application.

As mentioned earlier, evaluation should take place throughout the development cycle of the hypermedia application, as the cost and effort to correct errors, or to modify the application, can increase towards the end of the cycle. Unfortunately, as scale and complexity vary from system to system, there is no prescribed evaluation rationale on when or how many times an evaluation should be conducted. Nevertheless, evaluations should be conducted at least twice, once during the design stage (formative) and once when the application is nearly completed (summative) [Faulkner, 2000]. Obviously, formative and summative evaluations can be repeated several times, assuming the availability of all the necessary resources.

4.7 Summary

The first part of this chapter presented a number of formal methodologies for hypermedia applications that have been widely discussed in the hypermedia literature. These methodologies are required to ensure the consistency and structure of the application, and hence will help reduce maintenance and updating effort. Nevertheless, the most important parts of the hypermedia design, namely *aesthetic* and *cognitive* issues, are not supported by the formal design methodologies. Therefore, in designing a task support system, design rationale needs to be considered to increase *coherence* and to reduce *cognitive overhead.* The development environment should support the developer's incremental and recursive activities through the creation of a fast experimental feedback loop, cloning tools and prototyping.

The second part of this chapter discussed the success models and evaluation methods extracted from both the human-computer interaction and MIS literature. The hypermedia usability model [Nielsen, 1995] incorporates both the end users and the authors as the target of the usability evaluation. By understanding the success criteria, the model can be use to evaluate the proposed development methodology for an industrial task support system.

5 Proposed development methodology and success model

This chapter describes Stage 2 of the research programme. The objective is to propose a hypermedia development methodology and a success model for an industrial hypermedia task support system. The proposed methodology comprises a number of steps representing a full development cycle of a hypermedia application, from requirement specification to systems maintenance. The methodology was used to develop a number of task support systems for the sponsor. The proposed success model, used to evaluate the industrial hypermedia task support systems, was deduced from existing models described in Chapter 4. It focused on the stakeholders of the development process, including the *end users* and the *manufacturing process* experts.

5.1 The need for a new development methodology

The literature survey on hypermedia development methodologies in Chapter 4 indicated that, despite the availability of a large number of methodologies to develop hypermedia applications, most are categorised as *design methodologies* as opposed to *development methodologies*. This is because they only focus on the design stage within the entire development life cycle of the hypermedia application. The design methodologies have followed two formal design techniques: Entity-Relationships modelling [Chen, 1976] and Object-Oriented modelling [Booch, 1994]. Consequently, the large number of existing design methodologies, all evolved RMM and OOHDM.

Unlike the design methodologies, most of the development methodologies located in the hypermedia and software engineering literature are based on the waterfall model [Royce, 1970] and spiral model [Boehm, 1988] with modifications to suit the application and their problem domains. Such examples include the methodologies proposed by Powell *et al* [1998], Lowe and Hall [1999], Crowder *et al* [2000], Olsina [1998] and Jacobson *et al* [1999].

Comparison of both design and development methodologies (see section 4.3) has suggested that no single development methodology is appropriate to be adopted 'as-is', without further adjustments to suit the industrial hypermedia applications. The reasons for this observation include:

1. With the exception of RUP [Jacobson *et al*, 1999], existing development methodologies lack detail on *how* each step should be carried out, *what* tools or techniques are required, and *who* should take responsibility for each step. As industrial hypermedia task support system will be wholly owned and managed by people in the factory (not by the hypermedia developers), a more comprehensive, step-by-step procedure on how to use the methodology to maintain and create content is required.

2. In most of the existing methodologies, there is a gap between the conceptual modelling of the application domain at the design stage and the implementation stage. A methodology may have a detailed description of conceptual modelling

of entity and navigation during the design stage, but lack support at the implementation stage.

3. There is also a gap between the techniques used in existing methodologies and the skills available in industry to implement these techniques. Many of the methods, for example WebML [Ceri *et al*, 2002] and RUP [Jacobson *et al*, 1999], originate from computer science research. As a result, industrialists may have to familiarise themselves with those techniques before they can use the related methodology. On the other hand, some methodologies are well known, but may be too general for an industrial hypermedia task support system.

4. The formal design and development methodologies (mainly from hypermedia and software engineering literature) do not incorporate human factors that may influence the success of an industrial hypermedia task support system. Therefore, methodologies from human-computer interaction (also known as the informal approach) need to be incorporated into the formal methodologies.

A new methodology is therefore required to address the above issues. The methodology will be based on an existing methodology from the literature. The selection criteria take into account the following issues:

1. *Familiarity* implies the level of knowledge or skill required by the hypermedia developers (particularly in the sponsoring company) to use the methodology.

2. *Applicability* is the extent to which the methodology is relevant to the sponsoring company and to other industries.

3. *Compatibility* is how well the methodology matches the hardware, software and other IT systems, without a major system upgrade.

4. *Maturity* of the methodology avoids the development process from frequent changes due to, for instance, standardisation.

5. *Availability* of the supporting conceptual notations and tools. This is crucial, particularly when selecting the design methodology.

6. *Completeness* or a level of detail is required to provide the detailed procedure of the development methodology.

Table 9 lists the selection criteria of the methodologies that will be the basis for the proposed development methodology.

Table 9 – Selection criteria for development and design methodologies

DEVELOPMENT Methodology	*Familiarity*	*Applicability*	*Compatibility*	*Maturity*	*Availability*	*Level of detail*	*Score*
Modified waterfall [Powell, 1998]	●●●●●	●●●○○	●●●○○	●●●●●	●●●○○	●●○○○	21
Engineering approach [Lowe & Hall, 1999]	●●●●○	●●●○○	●●●○○	●●●○○	●●●○○	●●●●○	20
IHDM [Crowder *et al*, 2000]	●●●○○	●●●○○	●●○○○	●●●○○	●●●○○	●●●○○	17
HFPM [Olsina, 1998]	●●○○○	●●●○○	●●○○○	●●○○○	●●○○○	●●●●●	16
RUP [Jacobson *et al*, 1999]	●●●●○	●●○○○	●●○○○	●●●●○	●●●○○	●●●●○	19

DESIGN Methodology	*Familiarity*	*Applicability*	*Compatibility*	*Maturity*	*Availability*	*Level of detail*	*Score*
HDM [Garzotto *et al*, 1995]	●●●○○	●○○○○	●●○○○	●●●●●	●○○○○	●●○○○	14
RMM [Isakowitz *et al*, 1995]	●●●●○	●●●●○	●●●●○	●●●○○	●○○○○	●●●○○	19
OOHDM [Schwabe & Rossi, 1995]	●●●●○	●●○○○	●●○○○	●●●○○	●○○○○	●●●●○	16
WebML [Ceri *et al*, 2003]	●●○○○	●●●●○	●●○○○	●○○○○	●●●○○	●●●●●	17

The strategy adopted to propose a new development methodology is as follows:

1. *Selecting the development methodology from existing methodologies.* Hypermedia literature has given only four options to the development methodologies. Choosing an existing methodology that is almost suitable, then adding new features to the selected methodology will be much more efficient than creating a completely new one.

2. *Adopting a formal design methodology.* Similarly, features from selected design methodologies will be selected and adapted to suit the industrial task support requirements.

3. *Combining formal with informal design methodology.* The literature shows that formal and informal methodologies have so far been used separately. However, both formal and informal methods need to be combined when developing an industrial task support system [Nanard and Nanard, 1995].

4. *Formulating a new methodology.* Having completed the above, a new methodology is then proposed.

The following section discusses the proposed development methodology for industrial task support systems in more detail.

5.2 Proposed development methodology

When hypermedia technology is used to develop an industrial task support system, other important elements, particularly human factors, need to be incorporated into the development phase. Thüring *et al* [1995] propose a design rationale to improve comprehension of the application by increasing *application coherence* and reducing *cognitive overhead.* Nanard and Nanard [1995] suggest that the design process of a hypermedia application, particularly one to be used as a task support system, should incorporate both formal and informal methods. This will allow the hypermedia developer to iteratively design and modify the application, using a fast experimental feedback mechanism.

The proposed development methodology is illustrated graphically in Figure 24. As with existing development methodologies, the steps to a large extent follow the *modified waterfall* model, also known as the semi-structured System Development Life Cycle (SDLC), primarily due to its maturity and familiarity. In particular, SDLC is compatible with existing IT infrastructure at the sponsoring company, thus ensuring the applicability of the methodology. Recent studies by [Britton *et al*, 1997] and [Barry and Lang, 2001] also supported this by revealing that most developers use the SDLC model for multimedia (or hypermedia) development, particularly within general industry.

In addition, to a large extent, the methodology also adopts the iterative cycle following both the spiral model [Boehm, 1988] and RUP [Jacobson *et al*, 1999]. Each step that forms the overall methodology will be described in the following sections.

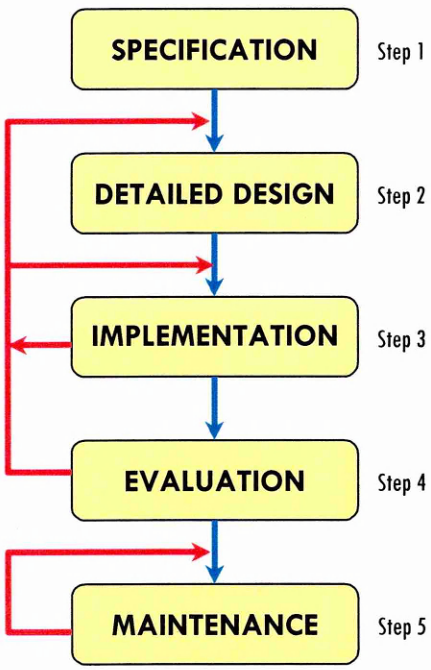


Figure 24 – Proposed development methodology

5.2.1 Step 1: Specification

A usable hypermedia task support system is determined by to the extent to which it enables a user to accomplish the task effectively and efficiently. The achievement of these goals will therefore depend on how the application reflects the user's working environment so that they are comfortable with the concept, terminology and process. More importantly, the application should illustrate the real task and use language and illustration that are familiar to the user. Therefore, it is important to understand how users perform their task. Previous research has demonstrated that successful design depends on a close collaboration between designers and users [Fakun, 2000; Ince and Kay, 1998]. Designers who spend time with users, understanding who they are and observing how they work are most likely to be successful [Hackos and Redish, 1998].

Before drawing a set of specifications for the hypermedia task support system, the task requirement has to be fully understood and to be matched with the user requirements. Task analysis can be used to describe how the task is carried out. There are numerous task analysis techniques, although Hierarchical Task Analysis (HTA) [Kirwan and Ainsworth, 1992] is one of the most popular techniques used to elicit task requirements.

HTA is a process of developing a description of tasks that need to be performed, and a statement of conditions necessary to undertake the task. HTA can assist the developer of a task support system to deal with cognitive aspects of the task and, therefore, it is ideal for designing operator manuals and job aids or even work organisation [Kirwan and Ainsworth, 1992].

Task analysis is particularly useful for developers who are not familiar with a task, as it helps them to interact with the prospective users of the system and to learn about the users by observing them performing the task. It is fundamentally different from asking the users what they do, because the users may not be able to articulate this, particularly if they are very familiar with the task. Task analysis focuses on gathering the information and understanding deeply how users perform the task [Faulkner, 2000].

While task analysis aims to gather the information about the task to be supported, the purpose of user analysis is to establish a profile of the prospective users of the application. It is imperative that this aspect be incorporated into this step to ensure that the users can utilise the application, irrespective of the quality of the content. The user profile can be obtained through interview, observation or questionnaire. Table 10 lists the common aspects of users collected during user analysis.

Table 10 – Aspects of prospective users

Profile	*Description*
Background information	Age range, gender, educational background, computer literacy level, cultural background
Level of expertise	First time (novice), intermittent, frequent
Use of the application	Voluntary or mandatory
Job detail	Job description or job level, main tasks and responsibility

In addition to task analysis and user analysis, another systematic approach to identifying the requirements is the use of Quality Function Deployment (QFD) [Akao, 1990]. QFD was primarily developed to give details on design and manufacturing processes to meet customers' needs, although it has also been applied in non-manufacturing environments, such as hotels and airlines [Zairi and Youssef, 1995] as well as in IT-related products [Tan *et al*, 1998; Barnett and Raja, 1995]. The crucial step in QFD is to identify *what* the users want (this is also known as *customers' voice)*. This can be done using either quantitative or qualitative techniques or, usually, a combination of both. A questionnaire followed by a focused user session (brainstorming) can be used to collect the requirements. These requirements are to be ranked according to the level of importance. Weightings are displayed next to each customer's voice. The *requirements* are then translated into the corresponding '*hows*'. The latter are the technical requirements or detailed specification on how to achieve the former. The results from

both user analysis and task analysis will form the basis for the formation of a requirements specification, consisting of functional, content and technology specifications.

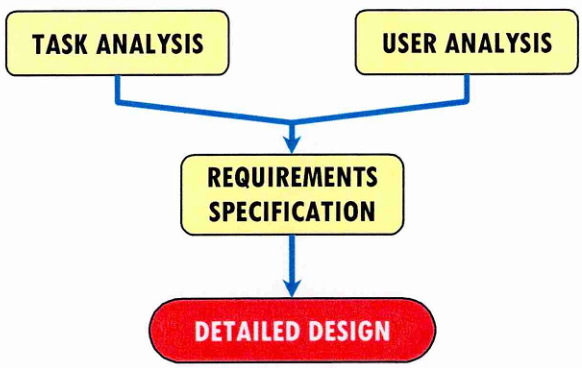


Figure 25 – Step1: Specification

5.2.2 Step 2: Detailed design

Nanard & Nanard [1995] remind the hypermedia developer that design is an incremental and recursive activity using both *formal* and *informal* approaches. The informal approach is essential, particularly when the hypermedia application is to be used as a task support system. It includes design heuristics suggested by HCI researchers (for example, Thüring *et al* [1995], Nielsen [1995], Gygi, [1991] and Fakun and Greenough [2002]), which can be incorporated into the formal design method.

Since hypermedia is used to support tasks that need deep understanding, as opposed to information retrieval, Thüring *et al* [1995] address the issue of designing for comprehension by incorporating two important factors: improving *coherence* of the hypermedia application and reducing *cognitive overhead* of the user. They propose eight design principles, which aim to address the above issues. Furthermore, the outcome from a study on hypermedia for maintenance by Koshy *et al* [1996] supported Thüring *et al* [1995] by recommending that hypermedia developers should carefully design navigation aids to help users reach the desired information with less effort and be extremely careful in controlling user disorientation and cognitive overhead.

Reducing cognitive overhead and improving coherence in a hypermedia application implies that the application has visual appeal, comprehensibility, navigability, and more importantly, that it supports the users' tasks [Shneiderman, 1997]. Thus, in addition to the eight design principles suggested by Thüring *et al* [1995], design heuristics for hypermedia applications pooled from the HCI literature are shown in Table 11. The first 4 groups concentrate on reducing the user's cognitive overhead whereas the last group deals with improving the coherence of the hypermedia application.

Table 11 – Design heuristics for hypermedia applications

1. Improve orientation	*Suggested by*	*Description*
Retrace mechanism / *breadcrumb* [1]	Thüring *et al* [1995]; Hardman [1995]	Help indicate user's current position
Tree structure, *Fisheye* view, Expand/contract	Nielsen [1995]; Lowe & Hall [1999]	Reconstruct the factors that led to this position
Book (tab) metaphor [2]	Fakun & Greenough [2002]; Lowe & Hall [1999]	

2. Facilitate navigation	*Suggested by*	*Description*
Guided tours	Nielsen [1995]; Thüring *et al* [1995]	Semi-linear navigation to guide novice user through information domain
Back/forward		
Open new window		Easy to return to original node after traversing to other nodes that are not directly associated

3. Node construction	*Suggested by*	*Description*
Chunked information or information clustering	Gygi [1991]; Nielsen [1995]	Reducing a large amount of text linking together a number of information units (as opposed to one long scrolled text)
Filtering/suppressing detail (view detail)	Gygi [1991]	Manage complexity by providing only information relevant to current context.
Illustration, animation or video clips	Mayer and Gallini [1990]	Improve comprehension by replacing a large amount of text with multimedia objects

4. User interface design		*Description*
Fixed screen size	Marcus *et al* [1995]; Thüring *et al* [1995]	Reducing effort in user interface adjustment
Fixed position of objects		Reducing effort in locating the user interface components
Split screen [3] **(frame)**	Lowe & Hall [1999]	Showing overall information context
Judicious use of colour	Fakun & Greenough [2002]	Increase speed of reaction
Sans serif font family (e.g. Arial, Tahoma)	Bernard *et al* [2001]; Tullis *et al* [1995]	Improve legibility, increase reading time

5. Improve coherence		*Description*
Tool tips	Lowe & Hall [1999]	Help the users to predict where the link will go, and hence the context of the link
Typed link labels	Gygi [1991]; Thüring *et al* [1995]	Hyperlink should be easily distinguishable from non-hyperlink
Overview diagram (site map)	Nielsen [1995]; Thüring *et al* [1995]; Gygi [1991]	Visualise structure of the application domain

Table 11 – Design heuristics for hypermedia applications (*continued*)

Note	
1	'Breadcrumb', also known as nodes labelling, allows the users to quickly locate where they are within the information domain and hence reduces the chance of getting 'lost in hyperspace'. The users can also trace the structure of the information, allowing them to return the location from which they started.
2	The design metaphor should be chosen according to the task supported by the hypermedia application. Metaphors are often used to relate the tasks being performed to something that the users are familiar with, particularly if taken from their working environment. Examples include the use of book metaphor in electronic manuals, in which the users can directly access the book's chapters using the tabs, or even linking one page with another, by using the program that mimics the page turning. Another example is using the picture of the workplace as the background of the application.
3	The screen is divided into a number of parts, usually not more than 3 parts, and some parts are used to access the information located on other parts. An example would be the use of frame in web pages.

The other axis of the design activity, according to Nanard and Nanard [1995], is the use of a formal design methodology. Among the many existing methodologies, HDM, OOHDM and RMM are the most frequently cited in the literature. Although HDM is a solid reference point for researcher in this field, it appears not to be used in practice [Lang and Barry, 2001]. This is because it concentrates more on the design representation or modelling rather than assisting the design process. This left two other options of design methodologies: one based on E-R modelling (RMM) and the other based on object-oriented modelling (OOHDM).

Both are equally popular and both have strengths and weaknesses. The selection of the technique in the detailed design of the industrial hypermedia system would therefore be influenced by a designer's experience or the ease of use of a technique, so as to reduce the learning effort required by the people in the industry. Another consideration would be the availability of a commercial tool that supports the technique, as this would speed up the design process.

At present, commercial tools to develop hypermedia applications are mostly supported by database integration capabilities using the well-known relationship database management (RDBMS). In addition, the E-R model has been recognised as a standard for the RDBMS, for at least the last two decades. The majority of systems analysts with experience in developing applications with relational databases would therefore likely be able to use such hypermedia tools. This led to the selection of E-R modelling technique and of RMM as a base methodology for the detailed design.

As mentioned in section 4.1.2, RMM addresses the design of hypermedia application through a seven-step process. However, in this research, user interface design will follow navigation design, whereas in the RMM, navigation design is followed by

conversion protocol design. The reason is that, when developing a hypermedia application, it would be easier to start from the user interface design. Furthermore, in RMM, user interface design, conversion protocol design and run time design are all sketched as three parallel steps following navigation design. Therefore, it can be assumed Isakowitz *et al* [1995] have given no predetermined sequence for those steps.

In addition, the conversion protocol design and the run time design are removed from the design step, and are now part of implementation step. This is because the design evaluation of the light prototype produced is required after the first four sub-steps, i.e. E-R design, slice design, navigation design and user interface design. This allows the developer and the prospective users to discuss various issues related to the design, including usability of the user interface.

In summary the detailed design consists of four sub-steps. They are **E-R design**, **slice design**, **navigation design** and **user interface design**. What distinguishes the detailed design in this research and the original RMM is that, at the end of user interface design, a *light prototype* is produced, as suggested by Nanard and Nanard [1995]. A feedback mechanism is attached to the detailed design through the **design evaluation**.

The modified RMM adopted in the detailed design phase is shown in Figure 26. Each part of the detailed design is described below, where steps are derived from RMM [Isakowitz *et al*, 1995] unless otherwise stated. The notations used in RMM are given in Appendix A.

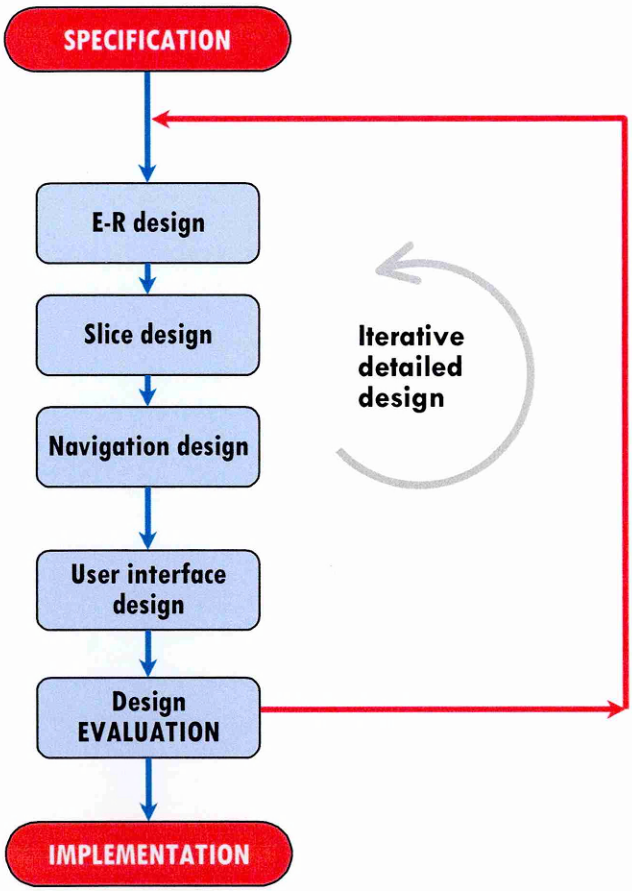


Figure 26 – Step 2: Detailed design

5.2.2.1 E-R design

This step aims to represent the information domain of the application via an Entity-Relationship (E-R) diagram [Chen, 1976]. It studies the entities and their relationships, which will form the basis of the hypermedia application as nodes and links. An analysis of the application domain will help the developer identify relationships between linkable entities which appropriate navigation can support. By defining associative relationship in the E-R design, a suitable navigational link can be created. Figure 27 illustrates a simplified example of the relationships between book, author, ISBN number and shelf number in a library information system. The result from this step is an E-R diagram for the entire application.

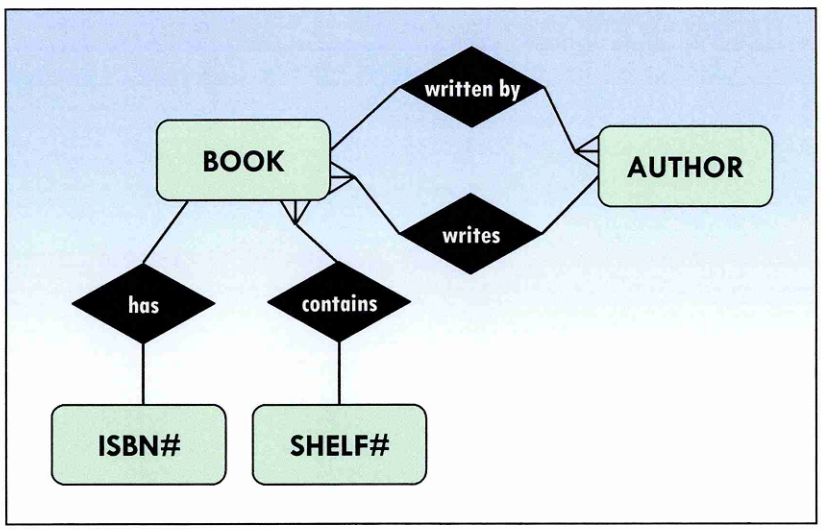


Figure 27 – Example of an E-R diagram

The above diagram explains that a book can only have one ISBN number (*one-to-one* relationship) and a shelf number may contain many books (*one-to-many* relationship). The relationship between book and author is that many authors may write a book and an author may write many books (*many-to-many* relationship).

5.2.2.2 Slice design

This step is unique to hypermedia applications and it does not exist within the usual software development process. The slice design determines how the information in the chosen entity will be presented to users and how they may access it. Here, an entity is split into meaningful slices and organised into a hypertext network. Each slice groups one or more attributes of an entity. This process is called the *slice design* phase and results in a *slice diagram*. The slice diagram also models the navigation between slices via unidirectional and bidirectional links.

Slice design is particularly useful during the interface design where an entity may be presentable within one 'window'. Although this approach is simple, users may become disoriented, as they have to scroll-down to find the information. With slices, an entity can be divided into meaningful units, which can be presented in multiple windows. Figure 28 shows an example of a slice diagram of the 'book' entity in Figure 27.

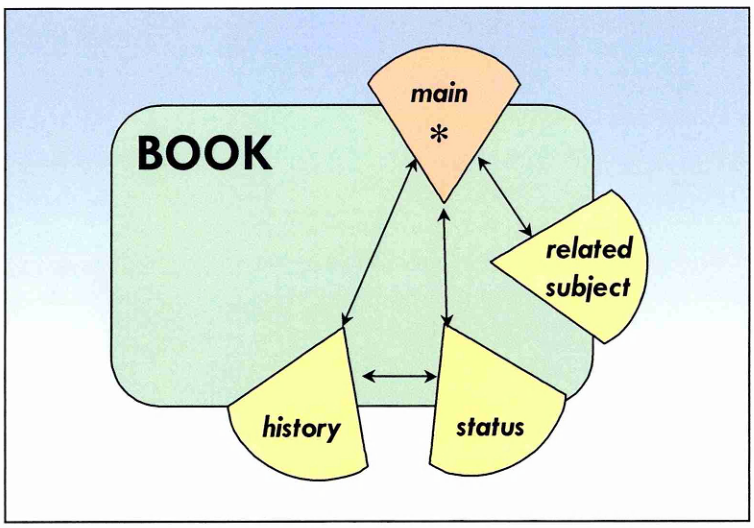


Figure 28 – Example of a slice diagram

The 'main' slice may contain the main information about a particular book such as title, author(s), ISBN number, shelf number and availability. The 'history' slice will also display title and ISBN number but, perhaps more importantly, the history of when this book was bought and to whom this book has been lent. The 'status' slice may present the title of the book and the detailed status of the title, such as that it is a reference copy or the number of copies held on the library. Finally, in the 'related subject' slice, details of the subjects or disciplines that relate to a particular title will be presented. The asterisk on the main slice, called the head of slice, indicates the default to anchor links coming to the book entity.

5.2.2.3 Navigation design

This step involves the design of navigation paths or hyperlinks. Each associative relationship between entities and each structural relationship between slices is analysed. RMM is particularly useful for applications that are updated on a relatively frequent basis. Therefore, all the hyperlinks are specified in generic terms and no hard coded links exist between instances of entities.

Figure 29 shows an example of the navigation diagram.

Here *groupings* are used to provide hierarchical access from the library main menu to the other sub-menus, *catalogue* and *user*. From the catalogue sub-menu a list of books of interest is displayed according to a particular search strategy, i.e. keyword, and each of the search results will be accessible through both *index* (list of books) and *guided-tour*. This is also the case when access is sought via the author's name. On the other hand, it is sometimes desirable to find out who has borrowed a particular book. The list of users who borrow the book will only be accessible by a *guided-tour*.

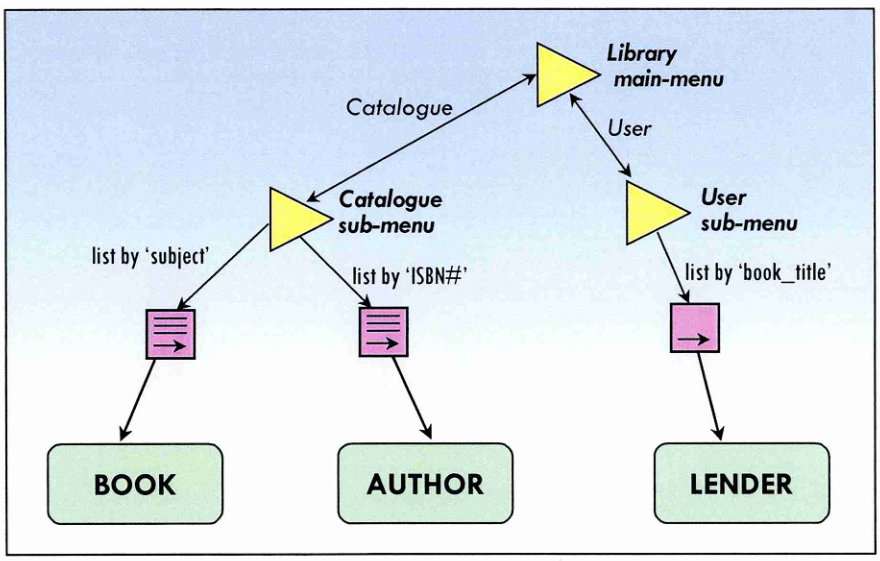


Figure 29 – Example of a navigation diagram

5.2.2.4 User interface design

The user interface design involves designing screen layout for every object that appears in E-R, slice and navigational designs. Design heuristics will be extensively used in this step to design the following user interface components:

- ***Main page layout***. The main page layout should provide an appropriate visual impact and enhance the usability of the application. Lowe and Hall [1999] suggested using design metaphor, split-screen and appropriate labelling of information nodes to ensure that the page layout design presents a consistent look and feel.

- ***Page layout for slice diagram***. In addition to the main page layout, user interface design also deals with the interface to display the information in the slice diagram. A typical example of slice layout is the 'detail page' found in many e-commerce web sites. When a user searches for a product, a list of hits is returned in tabular format with brief information about that particular product, for instance price and availability. A button is provided to open another window with more detail information about the specification.

- ***Access structures*** to link every object shown in the navigational diagram. This is related to the use of structural hyperlinks, where users are given access to sub-menus and to the entity through an access structure.

- ***Navigation aids*** (or buttons) for the guided tours. Very often a linear structure that links one node to another is provided, for example in encyclopaedia or in an electronic user guide. Therefore, buttons to facilitate the navigation have to be

designed to move forward to the next page and to go backward, as well as to the last or the first page.

The user interface can be designed using pencil and paper (the so called 'low fidelity prototype') [Rettig, 1994]. Another option is to use a computer to generate a 'high fidelity' prototype. Software packages such as Macromedia Fireworks and Adobe Image Ready can be used to generate the latter quickly.

Figure 30 shows a high fidelity user interface design that incorporates a design metaphor, the use of split screen and the bread-crumbing technique. It also shows the use of navigational aids and access structures.

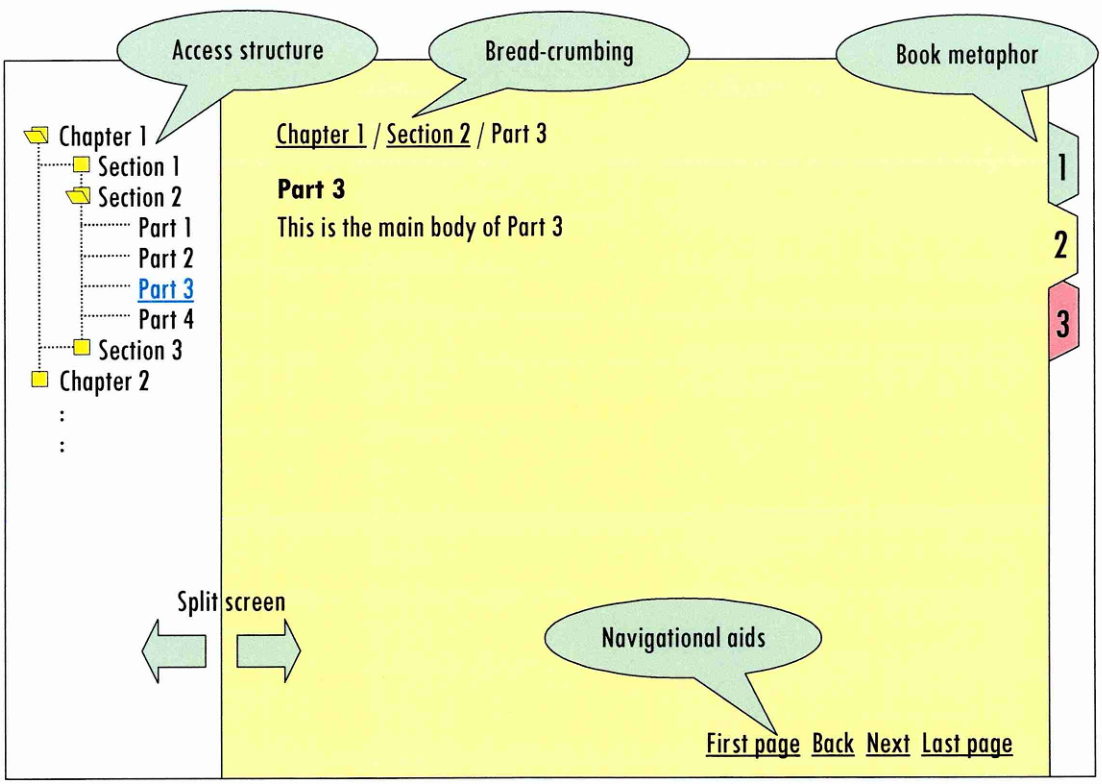


Figure 30 – User interface design using high fidelity prototype

5.2.2.5 Design evaluation

The aim of design evaluation is to get some early feedback on the usability of the design prototype. Prospective users with different levels of expertise are involved in a brainstorming session to review the prototype and to provide comment as to how it might be improved.

The session may start with a short demonstration. Potential users are then asked to give their opinion about what they like or dislike about the prototype and what changes they would like to make. Prototype evaluation before implementation will minimise costly

revisions when the application has been coded and completely developed. Therefore, investing in design evaluation is expected to save resources in the long term by minimising major modifications during implementation and after deployment. In theory, effective interaction between end users and the developer would improve the prototype.

Design evaluation also gathers the users' ideas as to which types of multimedia object need to be incorporated into the hypermedia application. The options range from a simple text/picture combinations to complex blends of rich media including text, still pictures, animations and digital video. The choices are determined by the requirements and the type of the task to be supported by the hypermedia application.

5.2.3 Step 3: Implementation

This is the step where the logical design and design prototype from the previous steps are converted into a physical, working prototype. Authoring software is required and this needs to be selected for the purpose, together with the chosen multimedia objects. This selection also depends on the delivery medium of the application. For instance, if the medium is CD ROM, the authoring software need not necessarily be web-based, which otherwise may require even more careful consideration due to bandwidth limitations and hence the multimedia objects used.

Once the authoring software and the multimedia objects are selected, the conversion protocol design and run time design steps are run. The former is necessary if the application involves structured data either in relational database or *flat file* format. Run time design is the actual physical programming of the application.

The following sections discuss both conversion protocol design and run time design, derived from RMM [Isakowitz *et al*, 1995] unless otherwise stated.

5.2.3.1 Conversion protocol design

If the hypermedia application requires a database backend, this step produces a set of rules that will be implemented as an instruction for converting E-R, slice and navigational designs into physical objects in the target hypermedia application. This includes translating the logical E-R diagram to a relational database's relationship and results in query design, for example using SQL language.

The ideal technique in this step is the availability of a Computer-aided Software Engineering (CASE) tool to automate the process. However, manual conversion can be done directly in most relational database management systems if a CASE tool is unavailable. Entities are converted into tables and attributes become fields (or columns), followed by the creation of the physical relationships between tables reflecting the relationships between entities. The next step is to convert the slice diagram into queries, where the indices are displayed as query results.

5.2.3.2 Run-time design

Run time design develops the skeleton of an application by building the physical user interface using the authoring software, as well as embedding codes (perhaps some programs or scripts) into the user interface. For instance, bread-crumbing navigation in web pages is usually programmed using client-side scripting with JavaScript, so it automatically creates the links on the fly, depending on the page that is loaded. The program will detect the structure of the page and create the corresponding structural links.

If a database is used, some codes are required to link the hypermedia application to the database, usually using an Open Database Connectivity (ODBC), to create the search algorithm or SQL queries and to finally to display the retrieved record(s) from the table. The hypermedia developer also needs to specify which parts of the information need to be generated dynamically, which parts of the information constitute links, and where these dynamic links go. Figure 31 shows the implementation steps.

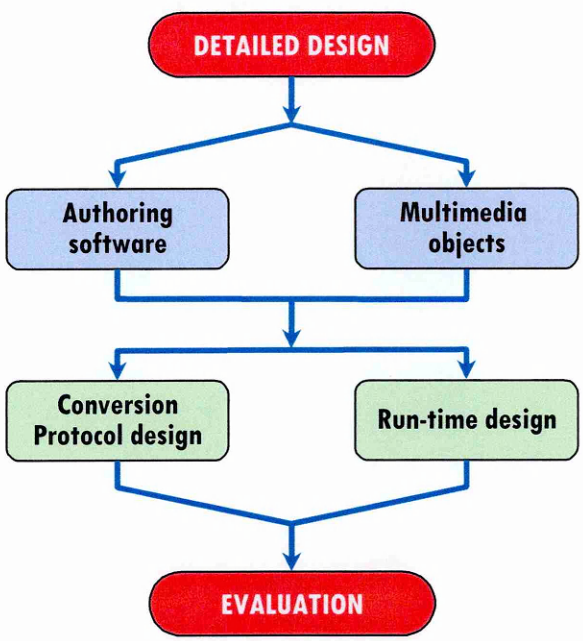


Figure 31 – Step 3: Implementation

5.2.4 Step 4: Evaluation

The purpose of this step is to refine the working prototype before it is deployed as a final application. Because an iterative design to refine the design prototype has been conducted in the former step, this step would be carried out in a more formal way to gather empirical data about the hypermedia application using a functional test and a usability evaluation. Functional tests evaluate whether the hypermedia application functions properly. The techniques are adopted from Mosley [1993], for example those

in automated dynamic testing category. Unit, integration and system tests are all conducted according to a test plan.

Once the functional testing is completed, depending on the objective and the scale of the hypermedia application, one or more evaluation methods can be chosen as a summative evaluation. User feedback is the qualitative method to elicit the users' opinions about the application. This may be followed by usability trials, in which the users are asked to use the application and are observed during different test scenarios. Any problems that may occur should be recorded. Quantitative data is collected through questionnaires or user surveys, using the proposed success model discussed in the next section.

Evaluation is an iterative process where the prototype is continually modified until it satisfies the design requirements. The results of design evaluation are fed back to either the detailed design step or the implementation step, depending on the degree of modification required. If no major problems are detected, then the hypermedia application is ready to be deployed. It is, however, possible that an evaluation might reveal problems unforeseen by the developer, which would lead to a redesign.

Figure 32 shows the proposed evaluation step of the hypermedia application.

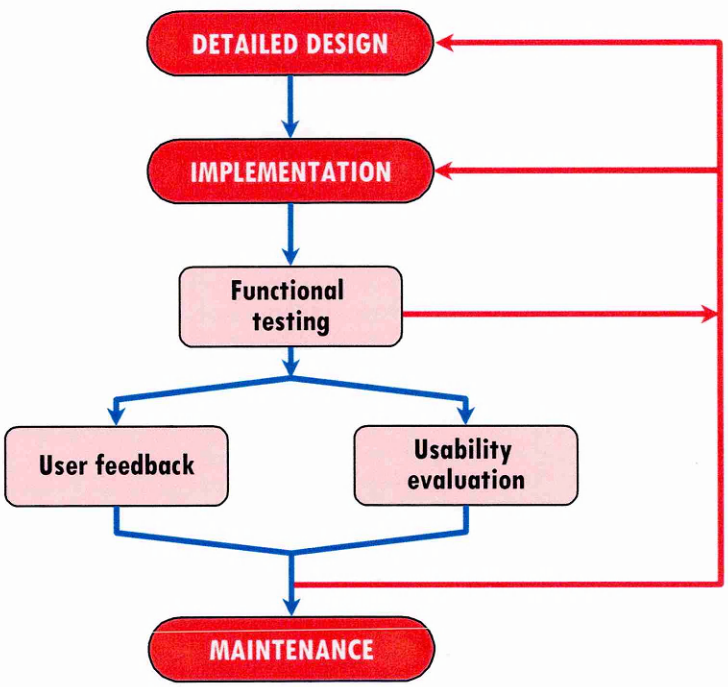


Figure 32 – Step 4: Evaluation

5.2.5 Step 5: Deployment and maintenance

A prototype that has been successfully tested and evaluated can then be deployed to the end users. If the application is delivered using CD ROM, then it has to be authored in

such a manner that the application can work properly, particularly if it needs installation on the client's PC. If the application is delivered using the web or intranet, it must be uploaded to a web server. Whether the application is CD-ROM-based or web-based, the content must be integrated and indexed for quicker information retrieval.

Various techniques, automatic link checker for instance, can be employed to check the hyperlinks. This feature is built into today's HTML editor and web development tools. Providing that there are no broken links, the application can now be released for publication.

Maintenance is also an on going process where the content, and particularly the links, are continually monitored. Updates have to be done to ensure that the content remains accurate and up to date.

Figure 33 shows the deployment and maintenance step of hypermedia application.

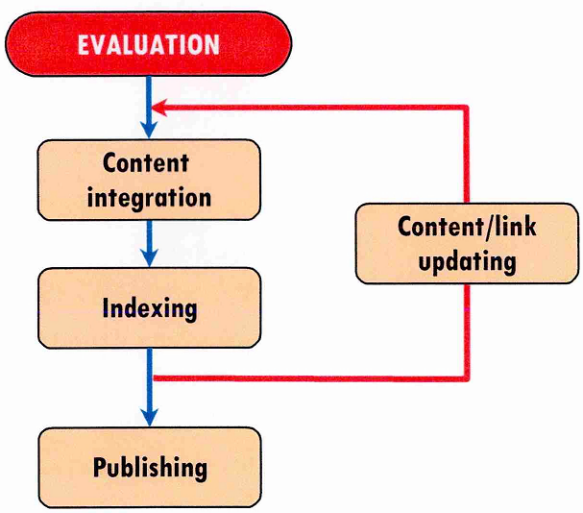


Figure 33 – Step 5: Deployment and maintenance

Figure 34 illustrates the complete development methodology described in sections 5.2.1 to 5.2.5.

Table 12 shows the development methodology using tabular format, where each step within the methodology is listed along with the processes, input, output and the tools or techniques used.

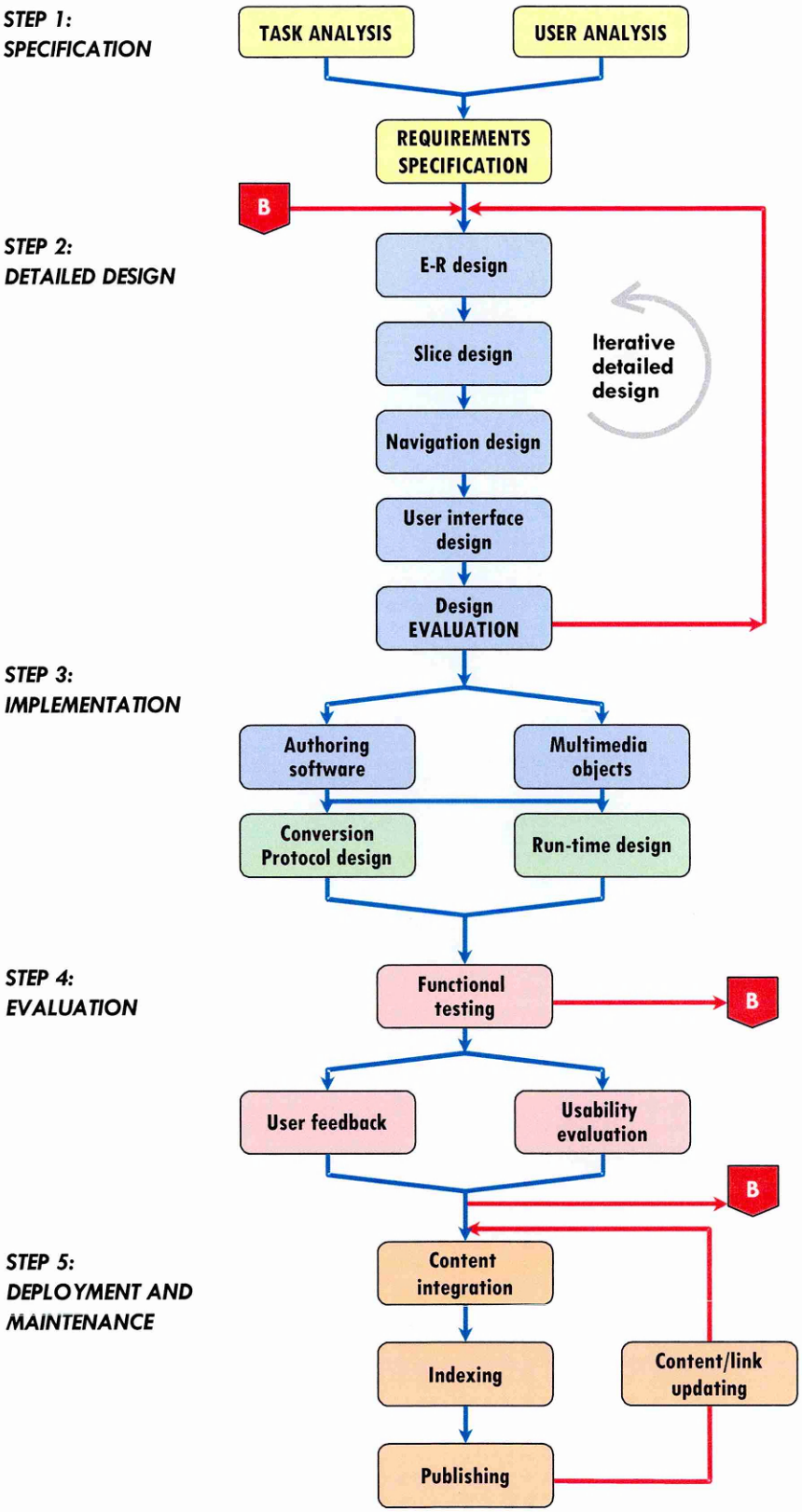


Figure 34 – The proposed development methodology

Table 12 – Proposed development methodology

Step	Input	Processes	Output	Tools & techniques
1. SPECIFICATION	Problems /requirements	1. Task analysis 2. User analysis 3. Technology analysis	Design specification	-Hierarchical Task Analysis (HTA) -Quality Function Deployment (QFD) -Interview, brainstorming -Scenario analysis
2. DETAILED DESIGN	Design specification	1. Information design ▪ E-R design ▪ Slice design	E-R diagram Slice diagram	-Commercial off-the shelf tools, such as Visio, FlowCharter, CASE tools -RDBMS
	E-R diagram Slice diagram	2. Navigation design ▪ Navigation model	User interface diagram	- Commercial off-the shelf tools, such as Visio, FlowCharter
	E-R diagram Slice diagram Navigation diagram	3. User interface design ▪ Design heuristics	Low fidelity prototype	-Storyboarding, design metaphor, -bread-crumbing, framing
3. IMPLEMENTATION	Design specification	1. Selecting authoring software	Skill requirements, cost, training needs, compatibility	
	Design specification	2. Choosing type of multimedia objects used	Skill requirements, cost, training needs, compatibility	

Table 12 – Proposed development methodology (*continued*)

Step	Input	Processes	Output	Tools & techniques
	E-R diagram Slice diagram Navigation diagram User interface diagram	3. Protocol design ▪ Table & query design	High fidelity prototype	-CASE tools, RDBMS
	Low fidelity prototype	4. Run-time design ▪ Programming	High fidelity prototype (feedback to DESIGN)	- Commercial off-the shelf tools, such as FrontPage, Visual Studio, DreamWeaver Ultradev, ColdFusion, Visual Basic, Flash, Director
4. EVALUATION	High fidelity prototype	1. Functionality testing (design-oriented	High fidelity prototype (feedback to DESIGN)	Walkthrough
	High fidelity prototype	2. Usability testing (user-oriented) ▪ Heuristic evaluation ▪ Usability trials & evaluation	Final application	Hypermedia success model TTF, TAM Interview (feedback session) Questionnaire
5. DEPLOYMENT & MAINTENANCE	Final application	1. Content integration 2. Indexing 3. Updating 4. Publishing	Final application	Indexer, link checker

5.3 Success model for industrial hypermedia TSS

In addition to the development methodology, a success model that was used during evaluation of the task support systems developed for the sponsor is proposed (Figure 35). To a certain extent, the proposed model is adopted from existing models from both MIS and HCI literature, and it focuses on the stakeholders of the hypermedia *development process*, including the *end users* and the *manufacturing process experts*. This model can be used to develop a set of success measures that so far do not exist for industrial hypermedia task support systems. Therefore, the aspect of usability evaluation through the adoption of various success models is deemed to be applicable to hypermedia task support systems.

As with TAM [Davis, 1989], the model utilises *intention to use* (IU) to deduce the system's success. However, *intention to use* is employed in both the evaluation of hypermedia applications that have been in operation for some time (post-implementation) and the applications that have just been introduced to the users (pre-implementation). When IU is used during pre-implementation, it attempts to measures the intention of the users to use the application in the future. On the other hand, during post-implementation the IU describes the intention of the users to continue using the application. Once again, it should be highlighted that the proposed success model is an extension of TAM, which is a *predictive model* [Davis, 1989]. Therefore, TAM can be used to predict system acceptability, but not to diagnose problems and application errors [Fakun, 2000]. The feedback obtained from usability trials can therefore be used to overcome the problems with the application.

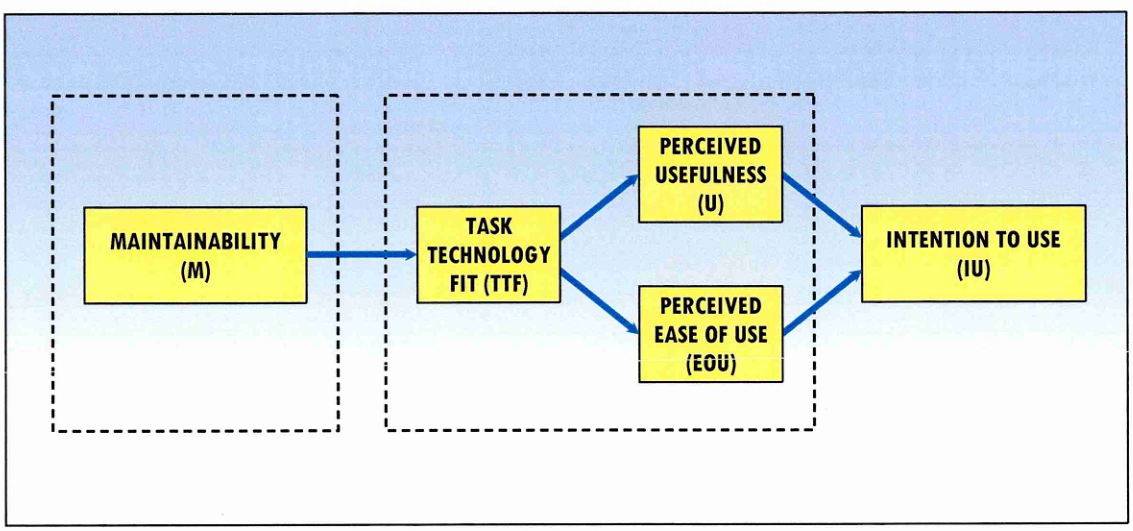


Figure 35 – Proposed success model

The validity of the success model is based on the following propositions:

1. The intention of the users to *use* and *continue using* the application (IU) are jointly determined by two constructs, *perceived usefulness* (U) and *perceived ease of use* (EOU).

2. Furthermore, both *perceived usefulness* (U) and *ease of use* (EOU) are determined by the degree to which the application fits the task it supports, or *Task-Technology Fit* (TTF)

3. Finally, the *Task-technology Fit* (TTF) depends on the *maintainability* of the hypermedia application.

5.4 Research instruments

Research instruments are required to collect empirical data to be used to evaluate the hypermedia task support system. An instrument (questionnaire) consists of a number of items, where one or more items are used to measure a construct. In this research, questionnaire items are selected from existing instruments that are reliable and valid, or formulated if necessary, to suit the industrial hypermedia applications.

To simplify the questionnaire, the hypermedia application that is evaluated will be referred to as 'the program'. All questions in the research instrument are designed using a Likert-type scale. The answers are fixed between 1 to 7, with 1 being 'strongly disagree' and 7 being 'strongly agree'. Fakun [2000] recommends not using too many questions or questions that are hard to understand, so that respondents are not confused or irritated. Therefore, a small set of items, chosen from a larger set of possible items found in the literature, will be used in this research.

5.4.1 Perceived ease of use (EOU)

The *perceived ease of use* (EOU) items are listed in Table 13.

Item 1 (*Learning to use the program is easy for me*) is selected because ease of learning is considered a substantial element of both EOU [Davis, 1989] and usability [Nielsen, 1995]. Item 2 (*I find it takes a lot of effort to become skilful at using the program*) is used to confirm that in using the hypermedia application, the users require minimal training to use the application. Item 3 (*Interacting with the program is frustrating for me*) is chosen because it is applicable to hypermedia applications where subjects are supposed to understand the navigation metaphor and the information space [Fakun, 2000]. Finally, item 4 (*Overall, I find the program is easy to use*) is chosen to explicitly emphasise the term '*ease of use*'.

Table 13 – Perceived ease of use (EOU) items

		Used in	
		Davis [1989]	Fakun [2000]
1.	Learning to use the program is easy for me	●	●
2.	It takes a lot of effort to become skilful at using the program	●	●
3.	Interacting with the program is frustrating for me		●
4.	Overall, I find the program is easy to use	●	●

5.4.2 Perceived usefulness (U)

The usefulness (U) items are listed in Table 14.

Item 1 (*I believe the program will make my job easier to complete*) is selected because the application should make job completion easier. Item 2 (*I think the program will support me to do my job*) is chosen since the users will find the applications critical to support their task. Item 3 (*I think the program will improve my job performance*) was not included by Fakun [2000], as he argued that his application was meant to be an information retrieval rather than a task support tool. However, item 3 is appropriate in this research. Finally, item 4 (*Overall, I think the program is useful*) is chosen to confirm the usefulness (U) of the application.

Table 14 – Perceived usefulness (U) items

		Used in	
		Davis [1989]	Fakun [2000]
1.	I believe the program will make my job easier to complete	●	●
2.	I think the program will support me to do my job		●
3.	I think the program will improve my job performance	●	
4.	Overall, I think the program is useful	●	●

5.4.3 Intention to use (IU)

Two items used by Venkatesh and Davis [1996] and Fakun [2000] are listed in Table 15. These items are adopted here to measure the *intention to use* (IU).

Table 15 – Intention to use (IU) items

		Modified from	
		Venkatesh & Davis [1996]	Fakun [2000]
1.	I intend to use [continue using] the program in the future, assuming I had access to it	●	●
2.	I predict that I would use [continue using] the program, given that I had access to it	●	●

5.4.4 Task Technology Fit (TTF)

The TTF instrument devised by Goodhue [1998] has 12 distinct dimensions. In his study, he argued that the dimension of fit applies to any IT system, including hardware, software and data. However, the TTF instrument would seem too specific to the information systems and services used to test it. As such, not all dimensions are appropriate for the evaluation of the industrial hypermedia applications in this research.

The four dimensions of TTF depicted in Table 16 should, theoretically, not be an issue in a hypermedia TSS, on the basis that the data are already linked in the system. The remaining 8 dimensions are considered relevant to measure the task-technology fit.

Table 16 – Dimensions dropped due to irrelevant context

Dimension	Definition
Locatability	Ease of determining what data is available and where
Assistance	Ease of getting help on problems with data
Compatibility	Ease with which data from different sources can be compared without inconsistency
Lack of confusion	A result of data stored in many forms and places

Furthermore, 3 dimensions are omitted from the remaining 8 dimensions. These are *level of detail*, *presentation* and *ease of use*. The first is dropped because it is crucial for the correct completion of the task. The last two are not used because of the presence of the TAM construct to measure *ease of use* of the application (including the presentation) through user interface testing. The remaining 5 dimensions of TTF used in the evaluation are shown in Table 17.

Table 17 – Dimensions measured and their respective definitions

Dimension	*Definition*
Currency (CUR)	The most up-to-date information
Meaning (MEA)	Ease of determining what the information or data means
Accessibility (ACC)	Easy access to desired information
System reliability (REL)	Dependability of access and uptime of systems
Accuracy (ACU)	Correctness of the information

Table 18 shows the items chosen to measure the 6 dimensions of task-technology fit depicted in Table 17.

Table 18 – Task Technology Fit items

		Suggested by		
		Goodhue [1998]	Dishaw and Strong [1998]	Goodhue and Thompson [1995]
CUR01	The information from the program is up to date	●	●	●
CUR02	I cannot get the information from the program that is current enough for my purpose	●	●	●
MEA01	I can understand easily the definition or the meaning of the data from the program		●	
MEA02	The exact meaning of the information I found from the program is obvious	●	●	●
ACC01	I can easily get the information to support my task from the program	●	●	
ACC02	It is easy to access the program and to get the information that I want	●	●	
REL01	I can count on the program to be up and available when I need it	●	●	●
REL02	The program is subject to frequent hardware and software problems and crashes	●	●	●
ACU01	The information is accurate enough for my purposes	●		●
ACU02	There are a number of errors and inaccurate information on the program	●		●

5.4.5 Maintainability (M)

Maintainability is defined as a set of attributes that bear on the effort needed to make specified modifications [BSI, 2001]. According to that document, there are four groups of attributes or dimensions that describe maintainability. They are:

1. *Analysability*: the effort needed for diagnosis of deficiencies or causes of failures, or for identification of parts to be modified.

2. *Changeability*: the effort needed for modification, fault removal or for environmental change.

3. *Stability*: the risk of unexpected effects of modifications.

4. *Testability*: the effort needed for validating the modified software.

In addition, it is important to measure how easily new content can be generated, when the hypermedia application has been deployed and in operation over a long period of time. Therefore, another dimension of maintainability, called *reusability*, needs to be appended to the aforementioned dimensions obtained from ISO/IEC 9126 [BSI, 2001]. *Reusability* measures the effort needed for creating or adding new content with a high degree of consistency of the overall look and feel of the application. Since no specific questionnaire items to measure maintainability could be located in the literature, these would be formulated on the basis of the above maintainability dimensions. The items are listed in Table 19.

Table 19 – Maintainability items

Dimension	Items
Reusability	It is easy to create new content that is consistent in look and feel
	User interface components can be easily reused to create new content
	The system requires minimal manual formatting (style is used)
	Less effort is required to manage the file (file naming convention is used)
Changeability	It is easy to update the content
	It is quick to update the links (handled by automatic hyperlink)
Analysability	It is easy to find faults, e.g. when a wrong picture is loaded
	It is easy to find broken links
	It is easy to find errors in document formatting
Stability	The system is not affected by advances in computer technology
	Multimedia objects do not require regular updates of the drivers/players
Testability	Changes can be tested and verified immediately

5.5 Summary

A review of hypermedia development methodologies suggested that no single development methodology is appropriate to be adopted 'as-is'; they all require further adjustments to suit industrial hypermedia task support systems. The deficiencies include: lack of detail as to how each step should be carried out; existing methodologies focus only on the end users; gaps between the conceptual modelling at the design step and the implementation step; gaps between the techniques used in existing methodologies and the skills typically available in industry; and the fact that existing methodologies do not take account of the human factors that influence the success of hypermedia task support systems.

A new development methodology is proposed to address the above issues. It adopts the semi-structured SDLC and spiral models, with many features being more detailed. The proposed methodology comprises five steps: specification, detailed design, implementation, evaluation and maintenance. Although WebML is the most detailed methodology, the tool is not yet available commercially[3]. Therefore, as suggested by Lowe and Hall [1999], the detailed design follows RMM [Isakowitz *et al*, 1995] with substantial enhancement, particularly in the user interface design. The latter adopts a design rationale and design heuristics obtained from HCI literature. Furthermore, the detailed design is a combination of formal and informal methods, iteratively carried out in close collaboration with prospective users. The development methodology provides a step-by-step guide that details the inputs, process, outputs, tools or techniques needed, as well as the individual(s) who will carry out each particular step.

In addition to the development methodology, a success model that can be used to evaluate the industrial hypermedia task support system has also been proposed. Like the success models found in the literature, it also adopts *use* as a surrogate measure for system success. However, the fundamental difference is that the proposed success model considers both the end users and the manufacturing process experts in the success criteria.

The model suggests that the successful application will depend on the end users' acceptance of the application, indicated by their intention to use or continue using the application. This is jointly determined by two other constructs: *perceived usefulness* and *perceived ease of use*. Furthermore, both *perceived usefulness* and *ease of use* are determined by the degree to which the application fits the task it supports (TTF). Finally, the TTF itself will be influenced by the maintainability of the hypermedia application, which may be indicated by the factors reusability, changeability, analysability, stability and testability. Thus, the provision of maintainable and usable industrial hypermedia applications is hypothesised to ensure the system's success.

[3] At the time the thesis is written, only a trial version is available.

6 Case studies

This chapter contains four case studies, which are related to the development and deployment of industrial hypermedia task support systems during the execution of the research project at the sponsoring company. The development of the hypermedia TSS using the proposed methodology, and the evaluation of the hypermedia TSS using the proposed success model, are discussed.

The first two cases discuss a hypermedia task support system called ManTIS (Manufacturing Test Information System) developed to assist the test personnel, including operators, technicians and engineers, to carry out a systems test of a base station. To accommodate the wide-range of the users' expertise, ManTIS is structured into three modules (also known as *modes* within Motorola). They are *simulation mode*, *reference mode* and *training mode*.

1. *Simulation mode*: is a learning tool aimed at new operators. The interactive nature of this mode allows the trainees to learn their tasks as if they were working with the real equipment.

2. *Reference mode*: is essentially an electronic handbook consisting of technical information relevant to the test procedures, for example wiring diagrams, data sheets and test specifications. The information is mainly aimed at engineers and technicians.

3. *Training mode*: has the same structure as the reference mode, but the information is less detailed and is presented step-by-step. The training mode is aimed at technicians and more experienced operators.

The ManTIS concept was used to develop two applications, called *Positrol help system* and *Multimedia MPS*, following ManTIS deployment on the shop floor.

Positrol, which stands for Positive Control, is part of the TL 9000 quality standard that obliges Motorola and other telecommunication companies to conform to the quality standards by regularly inspecting the test stations and peripherals before they are used to test the products, in order to ensure that the test results are trustworthy. To assist the operators to carry out Positrol tasks, an electronic help system, known as *Positrol help system* was developed.

A *Multimedia Manufacturing Process Specification* (MM-MPS) has been developed with the purpose of improving authoring process of the MPS; increasing management control of the documentation system; improving comprehension of current MPS and therefore ensuring the quality of task execution covered by the MPS.

The following sections describe the development of ManTIS, Positrol help system and Multimedia MPS. The discussion in this chapter follows the proposed development methodology presented in section 5.2.

6.1 Case 1: ManTIS simulation mode

Training methods for new test operators at Motorola were conducted on a face-to-face basis that would typically take a few weeks to complete. Motorola is a global company, and training for base station products is centred at the UK site. For those who come from other Motorola sites abroad, this can be costly.

Over the last few years, multimedia technology and the Internet have merged to form a new way of delivering training materials. The adoption of multimedia technology in the form of electronic simulation training using Internet or CD-ROM is therefore believed to cut training costs [Hasebrook, 1999]. Face-to-face training will be carried out only when the operators are required to be physically present to work with the real equipment.

ManTIS simulation mode is an interactive computer-assisted learning package tailored for shop floor operators learning how to test a base station. It is called 'simulation' because it actually simulates how the test procedure is carried out. The simulated testing environment itself consists of the test software, the base station, the components and the test station. The ManTIS simulation mode displays the testing environment on a computer screen, to enable the trainee operators to carry out testing procedures as if they were working in a real environment.

The ManTIS simulation mode was developed to serve two purposes: (1) as a training tool and (2) as task support or memory aid. As a training tool, the ManTIS simulation mode provides interactive, computer-based training materials for trainee operators. As a task support, it provides quick access to test procedures, delivered using a multimedia format.

6.1.1 Step 1: Specification

A large proportion of the systems test procedures are under the control of test software, which may prompt the operator to perform a number of actions, for instance to place or replace components in the base station or to turn the switches on and off repeatedly. The operator interacts with the test software by responding to an event triggered by another event. The test software then monitors the actions undertaken by the operator and – depending on whether the correct action has been chosen – the test software will either continue to the next step or provide further instructions to the operator.

The ManTIS simulation mode is therefore required to perform exactly the same sequence of tests and to prompt exactly the same messages to the users, who are mainly trainee operators. These features are particularly required in an off-line environment, i.e. when the trainee operator is away from the test station.

To develop the simulation mode, a Hierarchical Task Analysis (HTA) was carried out to fully understand the sequence of the test process. The results indicated that the most effective way to describe the sequence of the test process, and to simulate the test process, was to split the whole task into a number of sub-tasks.

The advantage of using this approach was that the users (trainee operators) would be able to learn the training materials step-by-step using modules. In addition, from the developer's point of view, it was easier to develop and to test the hypermedia application at its modular level [Crowder *et al*, 1999]. In ManTIS simulation mode, each sub-task represented a module and a group of modules were integrated to form the complete application. With modular application, the trainee operators could learn the tasks at their own pace, because it enabled them to repeat a particular sub-task without going through the whole sequence of tests. The simulation mode needed, therefore, to provide links from one sub-task to another, allowing the users to navigate the system in both sequential and non-sequential way.

As the application was designed to be self-explanatory, it was also found necessary to provide a brief introduction to the environment where the trainee would be working. The introduction included some basic material, such as the test station, tools and equipment, and a short explanation about the product to be tested. This needed to be followed by some material explaining the steps required before the test procedures took place, for instance: how to connect the base station under test to the test station.

In order to explain how the step was actually carried out in the real testing environment, a digital video clip, also known as a '*how-to-do video*', was required. This was very useful when the simulation module was operated in the online mode, along with the actual testing process. The trainee should be able to locate the components as illustrated by the digital video clip, and to take the necessary action.

Table 20 summarises the design requirements for the ManTIS simulation mode.

Table 20 – Requirements specification for ManTIS simulation mode

Introduction	
Application	ManTIS Simulation mode
Purpose	Training tool for testing procedures
Scope	Systems test area
Content	*Horizonmacro* test procedures
Responsibility	Test System Engineering (TSE)

Task characteristic	
Type	Task support; training tool
Main task	Initialisation and alarm test
Description	Functionality and security tests

User characteristic	
Job level	Trainee operator
Level of expertise	Novice to intermediate
Computing level	Infrequent to moderate Internet (WWW) users

Requirements	
Source of content	*Horizonmacro* MPS
Format	HTML
Access	From test station. Web browser ActiveX control
Multimedia	Flash animation, interactive animation, video clips
Delivery	Intranet (online) and CD-ROM (offline)
Retrieval type	Browsing
Access mode	Mainly sequential
Security	Read only
Entry points	Products list, e.g. HorizonMacro., MCell
Direct link	To ManTIS reference mode
Training	Minimal

6.1.2 Step 2: Detailed design

Based on the design requirements, an iterative detailed design, adopting both formal and informal methods, was conducted. Formal design was employed during the design of the information structure and navigation, whereas informal design, using the design heuristics described in section 5.2, was incorporated during the user interface design.

6.1.2.1 E-R design

The design requirements also identified four entities needed to construct the ManTIS simulation mode. They were: the task itself, the simulated test software, the digital video clip and the *virtual tools*. Each task corresponded to a simulated test software, which was an interactive animation comprising a series of screenshots of the test software. Therefore, the relationship between the task and the animation was *one-to-one*. Furthermore, each animation had more than one *digital video clip* demonstrating what needed to be done when the software prompted an instruction. Equally, each animation also corresponded to a set of *virtual tools*, where the operator could operate the tools using the computer. This implied a *one-to-many* relationship, from the animation to both digital video clips and virtual tools.

An E-R diagram, illustrating the relationship between the *task*, *animation*, *digital video clips* and *virtual tools*, is shown in Figure 36

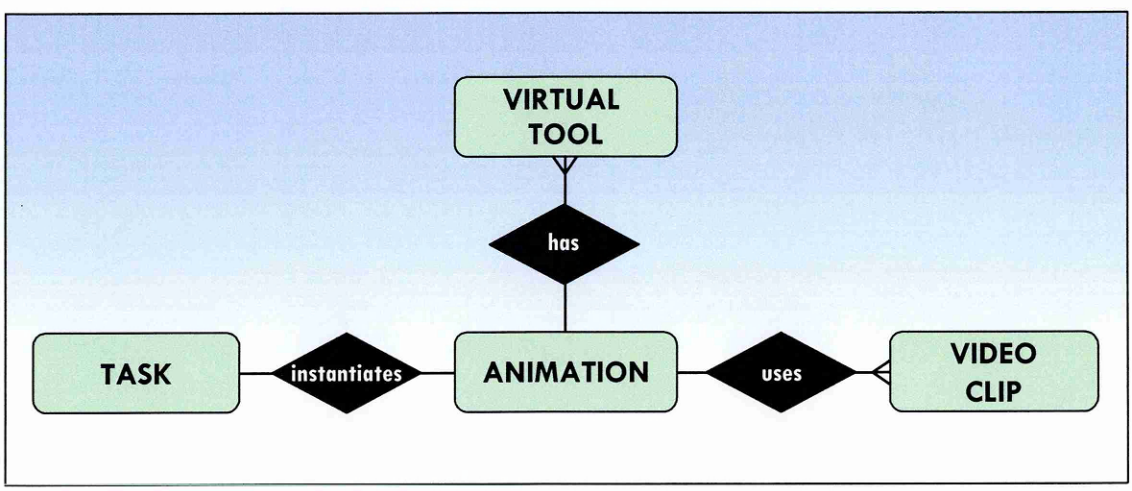


Figure 36 – E-R diagram of ManTIS simulation mode

6.1.2.2 Slice design

RMM [Isakowitz *et al*, 1995] defines a *slice* as one or more attributes that belong to an entity. The task entity, for example, has attributes such as task ID, task name or task description. Similarly, the video clip entity also has a number of attributes. Slice design involves grouping one or more attributes from one entity with attributes from another

entity, resulting in a slice diagram. The slice diagram is useful during the interface design, where slices determine how the information will be presented.

According to RMM notation, the smallest unit of a slice within an entity is called a *minimal* slice, for example the entity's name. During the slice design in the ManTIS simulation mode, a slice from the task entity was grouped with minimal slices from other entities to form an instance of the simulated test environment called *task_set*. During the use of simulation mode, different instances were presented simultaneously, depending on the task being simulated.

Figure 37 illustrates the slice diagram for the ManTIS simulation mode.

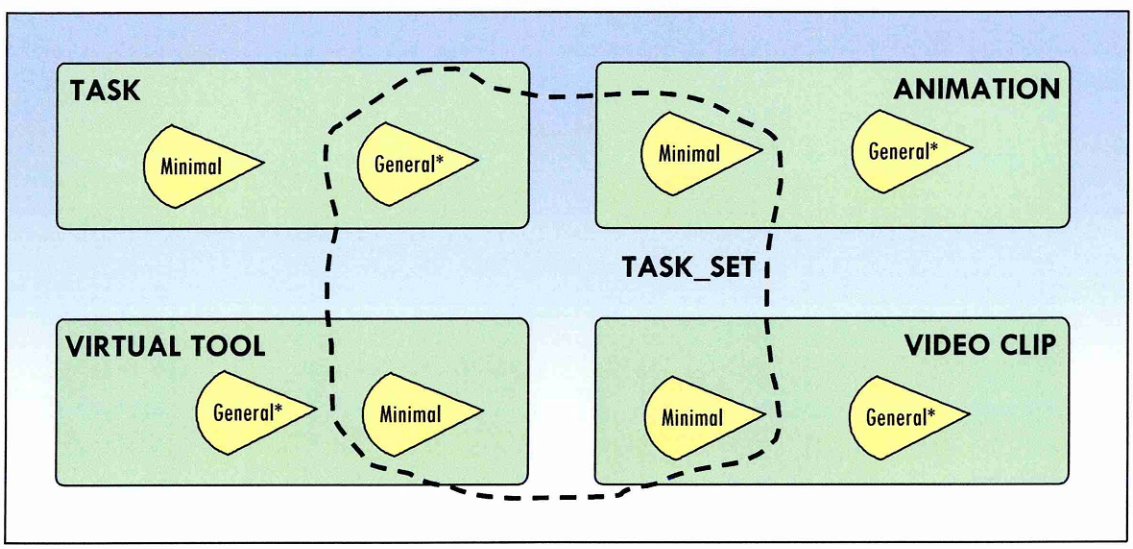


Figure 37 – Slice diagram of ManTIS simulation mode

6.1.2.3 Navigation design

Testing a base station is a combination of manual and automated tasks, which are controlled by the test software. The simulated test software in the ManTIS simulation mode also provides the test sequence and prompts the same commands to the operators to perform given actions. The default navigation type in simulation mode was therefore made sequential, because it was aimed at requiring trainee operators to follow the sequence given by the test software. Providing that the users performed the correct action at the correct time, the system would proceed sequentially to the next step.

As the users became more familiar with the test procedure, they might jump from one sub-task to another in a non-sequential way. Therefore, non-sequential access was also provided, so that they did not necessarily need go through the whole sequence of the test procedure.

A navigation diagram was produced, like the one depicted in Figure 38. Notice, from Figure 38, that there are two types of access from the task list to the *task_set*. The solid line represents non-sequential access, whereas the dashed line represents sequential access (also indicated by indexed guided tour).

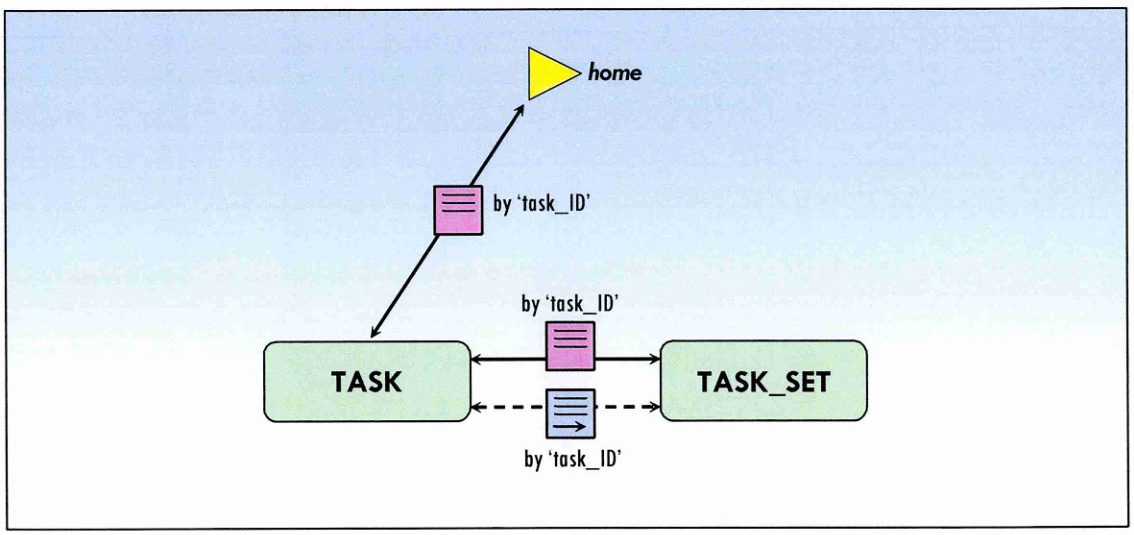


Figure 38 – Navigation diagram of ManTIS simulation mode

From the main entry point (*home*), a user can access the ManTIS simulation mode through a task list. Each task on the task list has a link to the associated instance or *task_set* indicated by the task ID. Thus the task ID is a unique identifier to retrieve the *task_set* and to populate the associated animation, digital video clip and virtual tools, into the user interface.

6.1.2.4 User interface design

During user interface design, the design heuristics in Table 21 were incorporated. As mentioned previously, animated test software was required to simulate an instance of the test software and hence to familiarise first time users with the test software before they actually went onto the shop floor. To simulate manual operations, various computer mouse interactions were required. Turning the switch on was simulated using a *click* of the mouse, whereas placing and replacing components was simulated using *click and drag*. A digital video clip was also presented to illustrate how those tasks were carried out using the real equipment.

The animation, virtual tools and digital video clip were laid out so that they would always be displayed at a fixed location (Figure 39). This format was used throughout the whole instance, and hence the whole task. The screen size was also fixed at full size to reduce the user interface adjustment effort [Thüring *et al*, 1995].

Table 21 – Design heuristics and how they were incorporated in ManTIS simulation mode

Design heuristics	*Applied*	*Description*
1.Improve orientation		
Retrace mechanism	No	-
Tree structure	Yes	Gives an overview of the sequence of the task and indicates current task or sub-task
Book/tab metaphor	No	-
2. Facilitate navigation		
Guided tours	Yes	Provided from within a task
Back/forward	Partial	Only forward and sequential navigation provided, so as to represent the nature of the test sequence.
Open new window	No	-
3. Node construction		
Chunked information	Yes	Task was split into a number of sub-tasks
Filtering/suppressing detail (view detail)	No	-
Illustration, animation or video clips	Yes	Each instance of the test process was represented by one simulated test software along with the digital video clip and the virtual tools
4. User interface design		
Fixed screen size	Yes	Opens in full screen
Fixed position of objects	Yes	Content frame is fixed into three parts to display simulated test software, digital video and virtual tools. No vertical or horizontal scroll bars are provided
Split screen (frame)	Yes	Three frames were used
Judicious use of colour	Partial	Considered only when selecting background and foreground colour (dark over light)
Sans serif font family (e.g. Arial, Tahoma)	Yes	-
5. Improve coherence		
Tool tips	Yes	'On mouse over' effects were used
Typed link labels	Yes	All links are underlined
Overview diagram (site map)	Yes	-

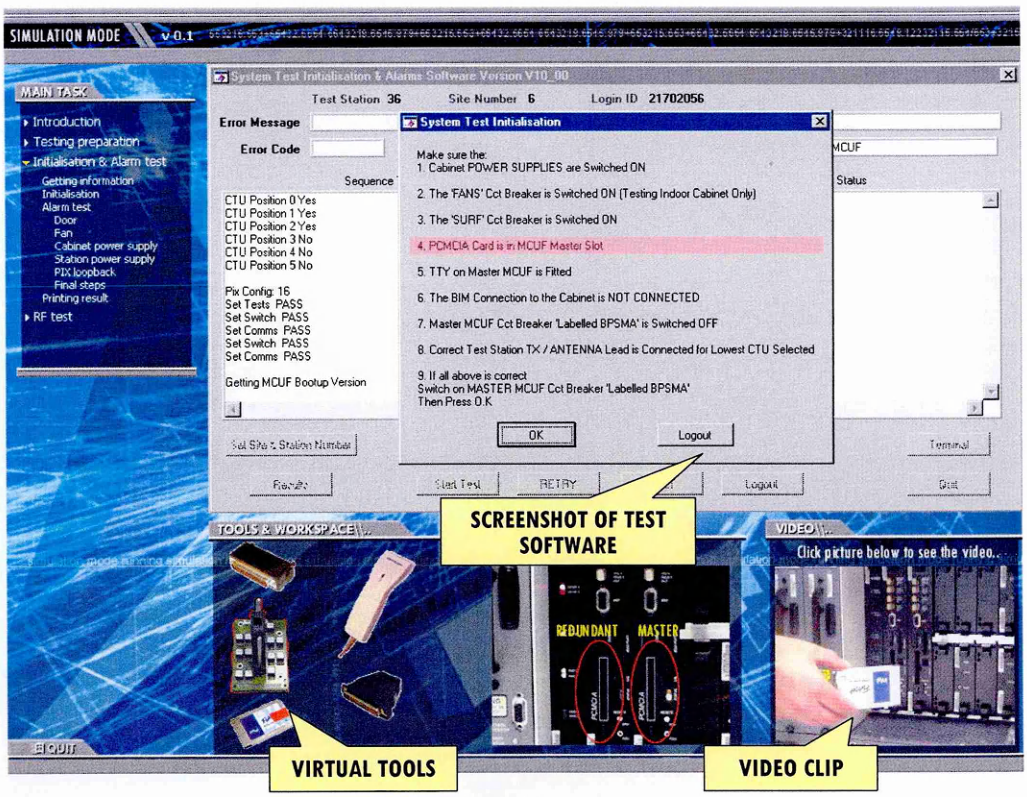


Figure 39 – User interface of ManTIS Simulation mode

6.1.2.5 Design evaluation

Design evaluation was conducted through a brainstorming session between the ManTIS developer and a number of manufacturing process experts, including technicians and engineers. The purpose was to verify the sequence of the ManTIS simulation mode, as well as to obtain feedback as to how each instance of the test process could best be supported by the digital video. In addition to verifying the functionality of the application, design evaluation also aimed to refine the user interface.

The results from the design evaluation indicated that, in addition to simulating the main task, it was necessary to include a number of preliminary tasks that needed to be carried out before commencing the main task, for example how to connect the base station's power supply, what connectors should be used, etc. Therefore, a number of extra tasks were added to the main task, including the video showing how to connect the base station to the test station.

6.1.3 Step 3: Implementation

This step was concerned with the selection of enabling technologies, and hence the required authoring tool. The ManTIS simulation mode was designed to operate in both online and offline environments. The former implied that the application was used as

task support during the task execution, whereas the latter implied that the application was used as a training tool. When used online, the simulation mode had to be accessible from any test station within the test area. Since all test stations were made Intranet compatible, the technology selected would be based on web and HTML. This is due to the fact that HTML has become a standard format for web applications [Silva *et al*, 1996]. In addition, HTML can easily integrate digital images and other multimedia objects. Moreover, it is cost effective, as the skill required to author HTML documents is relatively low, allowing changes without major development effort [Wu *et al*, 1999].

6.1.3.1 Conversion protocol design

Although an E-R diagram was produced during the iterative design, it was decided not to incorporate a relational database to store the content of the *task_set*. The main reason was that the ManTIS simulation mode relates to a generic product, not to a specific product configuration. This means that the content need not necessarily be updated regularly. Another reason was due to the delivery mechanism, i.e. CD ROM, implying the portability was very important. Having a database driven system would need the database to be installed before use, resulting in more effort and time to carry out that process. Therefore, the conversion protocol design in the ManTIS simulation mode was skipped.

6.1.3.2 Runtime design

To simulate the test software, screenshots of each stage of the test process were collected. The series of screenshots were then converted into animations that simulated particular instances of the test software. This was done using a multimedia-authoring tool called Macromedia Flash. Each animation controlled the logic of the ManTIS simulation mode. It judged the action taken by a user while using a virtual tool, for example positioning a virtual tool or clicking on the correct switch. Consequently, the virtual tool acted as the trigger of every event affecting the display of the screenshot of the test software, thus controlling the logic to the next sub-task.

The digital video clips were recorded with a digital camcorder and edited to reflect the task execution. The MPEG-4 compressor was used to compress the video, allowing substantial reduction in file size while maintaining a good quality of picture. ActiveX control was used to embed the video clip into the HTML page, allowing control buttons, such as *play* and *stop,* to be applied with a single click of the mouse button. Each video clip was synchronised with the task to be carried out; so the digital video in this case was considered to be an online help.

6.1.4 Step 4: Evaluation

The evaluation of the ManTIS simulation mode was conducted in two stages. The first stage was a user feedback session and the second was the usability evaluation. Unlike

the design evaluation carried out during the iterative detailed design, this step was carried out after the high fidelity prototype was developed. The evaluation method was therefore more formal, using a set of evaluation scenarios.

6.1.4.1 User feedback

Five separate feedback sessions were attended by five prospective users (hereafter called *subjects*). Two of these were experienced test operators. The remainder were operators from other areas who were not familiar with the systems test procedure. They represented trainee operators.

The sessions were organised to obtain feedback as to how the ManTIS simulation mode might be improved. Each session started with a short demonstration. During this demonstration, the subjects were informed that their suggestions would be helpful for improving both functionality and user friendliness.

After the demonstration, they were asked to interact with the application and to answer the following questions:

1. What do you like and dislike about the ManTIS simulation mode?
2. What would you change?
3. What is important to you when testing a base station, and what is not well simulated?
4. Would you recommend the application to your colleagues?

The answers to the above questions can be summarised as follows:

1. ***Preference.*** Subjects liked the idea of using a computer to learn their job. They believed that 'learning by doing' was the most effective way to familiarise themselves with their daily job. The simulation mode facilitated this by providing useful and more interesting learning materials than the conventional method. Nevertheless, as the technology was still new, there was a concern that the application could be too intense, as learning to use the application could take longer than learning to do their job.
2. ***Changes required.*** The majority of the subjects would have liked to have seen more digital video clips incorporated into the application. They also preferred a larger screen and hence the higher screen resolution, so that the screenshots of the test software could be positioned properly. Finally, they expected a quicker download time while maintaining a reasonably good quality of the video clips.
3. ***Important features.*** Since the ManTIS simulation mode was intended to be self-explanatory, it was suggested that novice users would require introductory sections describing the testing environment, tools and peripherals required. In addition, since not all manual operations could be simulated perfectly using

mouse actions, for example turning the *potentiometer* anti-clockwise, they preferred static pictures to illustrate these.

4. ***Recommendations***. The subjects would recommend the application to their colleagues, providing that certain changes they had requested were made.

6.1.4.2 Usability evaluation

The usability evaluation was carried out after the user feedback evaluation, by means of a questionnaire. It aimed to collect empirical evidence as to whether the subjects of the evaluation could use the application and whether they would continue to use it in the future. The subjects chosen for this evaluation were the systems test operators. The majority were experienced operators and hence, once again, it was decided to incorporate a number of operators from other areas of the assembly lines, to represent trainee operators who were not familiar with either the systems test process or the product to be tested.

The questionnaire was structured into two parts. The first part collected background information about the participants, including job level and task familiarity or task characteristic. The second part collected the perceptions about the ManTIS simulation mode, using the research instrument formulated in section 5.4. The constructs used were *perceived ease of use* (EOU), *perceived usefulness* (U), *task-technology fit* (TTF) and *intention to use* the system (IU). The items for each construct were fixed using a 7-point Likert scale, with 1 being 'strongly disagree' and 7 being 'strongly agree'. To simplify the question, 'ManTIS simulation mode' was referred to as 'the program'. A text box was provided at the end of the questionnaire for subjects who had additional comments.

The questionnaire used during usability evaluation is given in Appendix B.

Apart from participating in the feedback session, none of the subjects of the survey had used the ManTIS simulation mode before, neither had they seen the questionnaire prior to the usability evaluation. A total of 17 responses were collected but only 16 of them were used in data analysis, because many of the questions were left blank by one respondent.

The profile showed that 38% of the subjects were the inexperienced trainee operators and subjects from other departments (Group A), and the remaining 62% of the participants were experienced operators (Group B) who had been working in the test area for more than 7 months (Figure 40).

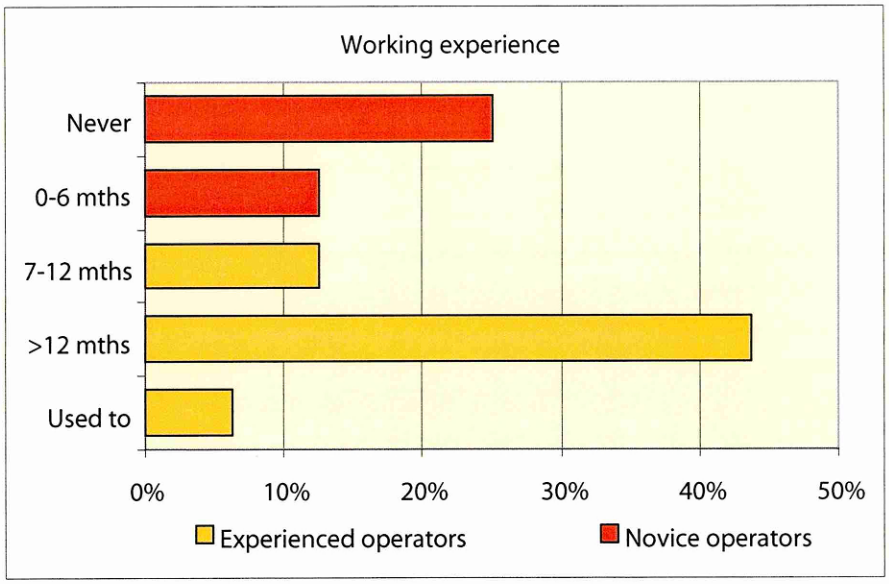


Figure 40 – Sample profile: working experience

Figure 41 shows the sample profile of task familiarity. The mean value of the task familiarity for Group A (novice operators) was 2.5 out of 7, implying that they were not familiar with either the systems test or the *Horizonmacro* product. On the other hand, the mean value for Group B (experienced operators) was 6.3 out of 7. In a further analysis, these will be used to investigate whether there is a correlation between these values and the values from other constructs.

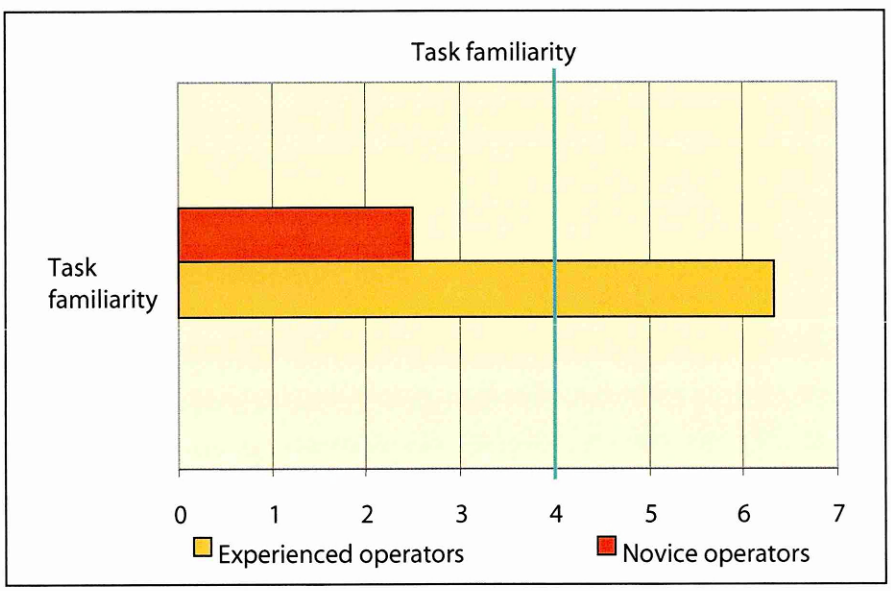


Figure 41 – Sample profile: task familiarity

Table 22 shows the indices of the four constructs collected by the questionnaire. The subjects perceived the ManTIS simulation mode as *easy to use* and *useful* as indicated by the high mean values of both EOU and U. They also indicated the *perceived TTF* (measuring the degree of fit between task and the tool to support the task) as moderately high. The mean of *intention to use* for the subjects was also high, showing their acceptance of ManTIS simulation mode.

Table 22 – Users' perception of ManTIS simulation mode

	Group A – Inexperience		*Group B – Experience*		*ALL*	
	Mean	SD	Mean	SD	Mean	SD
Perceived ease of use	5.89	0.61	6.57	0.56	6.25	0.63
Perceived usefulness	5.96	1.10	6.00	0.55	5.98	0.76
Perceived task-technology fit	5.61	0.98	6.30	0.56	6.04	0.8
Intention to use	5.25	1.04	6.05	0.88	5.75	0.97

Further analysis was undertaken to investigate whether *perceived ease of use* (EOU) and *perceived usefulness* (U) were affected by the task familiarity.

For each group, the mean value task familiarity was correlated against mean values of *perceived ease of use* (EOU) and *perceived usefulness* (U) (see Figure 42). For inexperience subjects (Group A), the graph shows no correlation between task familiarity and *perceived ease of use* (EOU). This means the *ease of use* of the ManTIS simulation mode did not depend on task familiarity. For the experienced subjects (Group B), there was a positive correlation between task familiarity and *perceived ease of use* (EOU). This was actually expected, since the subjects were familiar with their task and, consequently, they were able to predict how the application would respond to their interaction.

On the other hand, both groups indicated no correlation between *task familiarity* and *perceived usefulness* (U) (Figure 43). This was somewhat unexpected because, in theory, experience operators should have been able to judge whether the application was useful or would be useful in the future. Nevertheless, Table 22 shows that the operators perceived that the ManTIS simulation mode was useful, irrespective of their background experience.

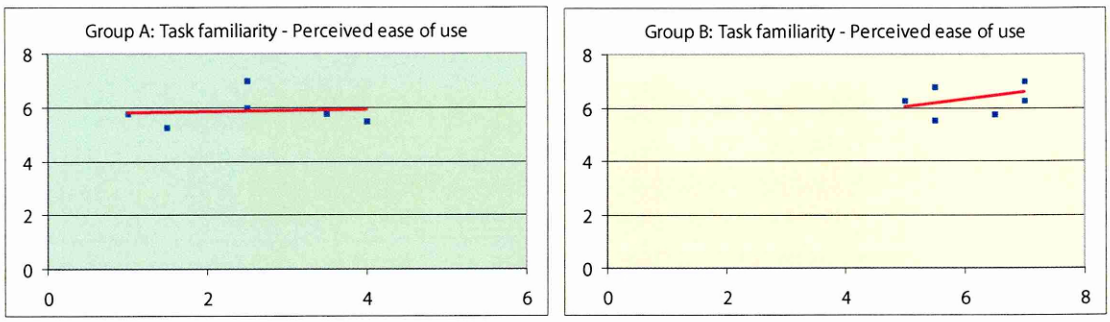


Figure 42 – The effect of task familiarity to *perceived ease of use* (EOU)

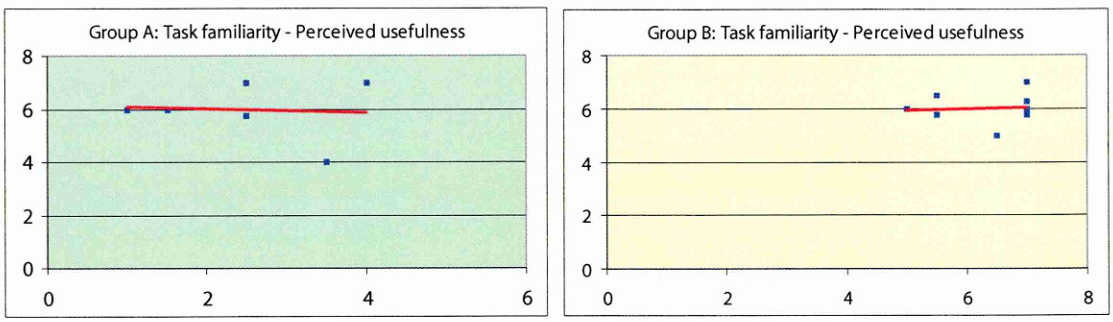


Figure 43 – The effect of task familiarity to *perceived usefulness* (U)

Another analysis undertaken was to investigate whether *task-technology fit* (TTF) had positive correlation to both *perceived ease of use* (EOU) and *perceived usefulness* (U); and whether *perceived ease of use* (EOU) and *perceived usefulness* (U) had positive correlation to the *intention to use* (IU).

Comparison between graphs (a) and (b) in Figure 44 suggests that that *task-technology fit* (TTF) is correlated to both *perceived usefulness* (U) and perceived ease of use (EOU), although the correlation on *perceived usefulness* (U) is slightly more significant than that on *perceived ease of use* (EOU). This is indicated by the steeper gradient of the curve in graph (a) than that in graph (b). Similarly, both *perceived usefulness* (U) and *perceived ease of use* (EOU) are correlated to the *intention to use* (IU) as shown in graph (c) and graph (d).

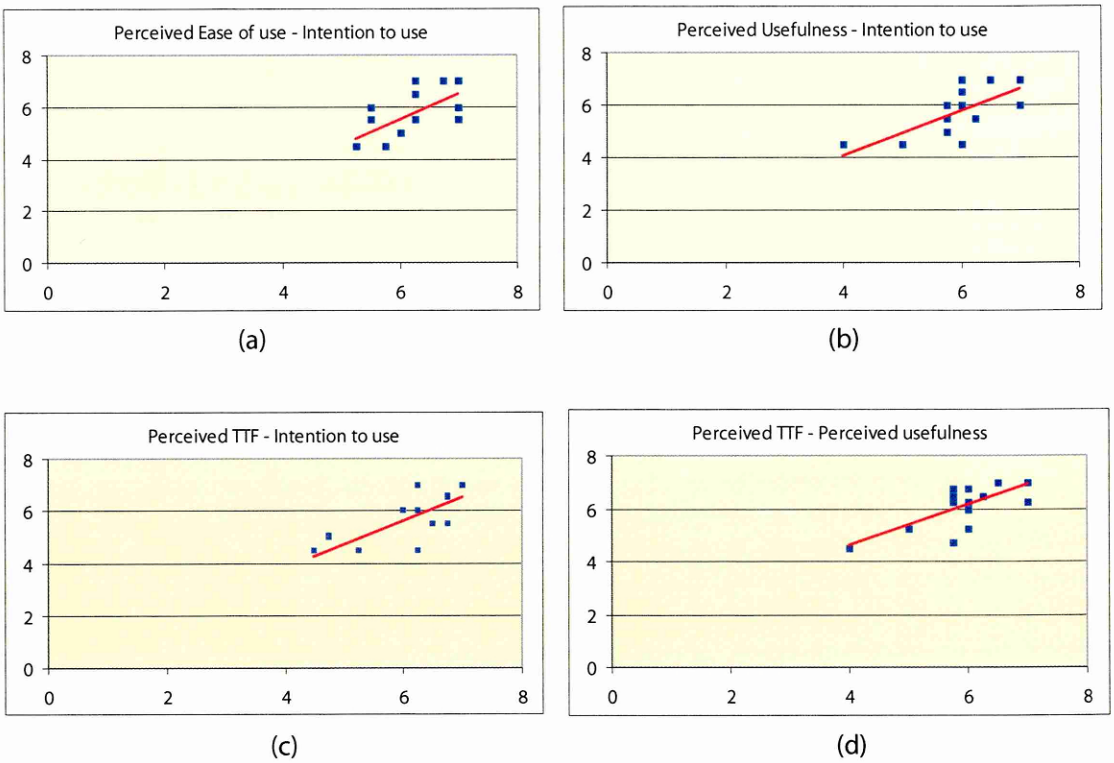


Figure 44 – Relationships between constructs

6.1.5 Step 5: Deployment and maintenance

Having conducted an evaluation, the ManTIS simulation mode was then deployed to the end users. It was first deployed as an offline application, delivered using CD ROM. The CD ROM had been used by both in-house test operators and a small number of Motorola partners abroad. The latest version was deployed through the company Intranet and was accessible from a number of test stations throughout the manufacturing facility. The IT administrator posted the HTML files along with the multimedia objects to the web server. Once the application was deployed, very little maintenance was required since the application would not be updated regularly. However, the methodology used to develop the ManTIS simulation mode for the *Horizonmacro* base station was utilised to create simulation modes for other products.

6.2 Case 2: ManTIS reference mode and training mode

The *ManTIS reference mode* is an electronic manual, consisting of technical information relevant to systems test procedures. The information is structured into sections and each section has smaller sub-sections to allow quick information retrieval. Because of the level of detail and the technical features within the ManTIS reference mode, it is primarily aimed at technicians and engineers.

The *ManTIS training mode* is a sub-set of the ManTIS reference mode, allowing less experienced users to access the technical materials found in the ManTIS reference mode in a simpler way. Compared to the ManTIS reference mode, the ManTIS training mode presents less detailed technical information. Each topic is presented step-by-step, using a predefined access route. Unlike the ManTIS reference mode, the ManTIS training mode is aimed at operators who wish to acquire more technical information regarding the test procedure, which otherwise would require more effort and time to understand, due to the complexity of the technology involved.

Both the ManTIS reference mode and the training mode are structured in such a way that they represent an associative relationship between *topics* of relevance. Within the sponsoring company, a topic could be a type of base station called *Horizonmacro*. The *Horizonmacro* has related attributes, for instance technology, features, test procedures, etc. Based on this relationship, reference mode and training mode associate the information from various sources into a logical, structured presentation of information to help the test personnel understand that topic.

6.2.1 Step 1: Specification

The word 'reference' implies that the ManTIS reference mode acts as an online reference library consisting of technical information related to the testing process. For that reason, the structure of the ManTIS reference mode needs to allow the users to determine the sequence of information retrieval that minimised time and effort. The structure is also required to enable the users to predict where the information could be obtained or, if they have accessed the information previously, to recall the location of the information. The latter is crucial when the ManTIS reference mode is used by frequent or more experienced users, who generally need a shortcut rather than a fixed access-route to obtain the same information.

In addition, the ManTIS reference mode needs to be accessible and linkable from other ManTIS modes. At the end of a predefined topic in the ManTIS training mode, users are offered a direct link to the related topic in the ManTIS reference mode, without having to browse through the entire content. This means that sections in the ManTIS reference mode have to be marked in such a way as to represent associative relationship with other sections of other ManTIS modes.

In the ManTIS training mode, diagrams and technical data were omitted and replaced by more user-friendly sentences with less jargon. Whenever necessary, more pictures were provided to explain complex processes, which otherwise would need a large amount of text. Users need to be guided forward and backward sequentially and therefore access structures (for example *guided tours*) were found suitable for this purpose.

As the application also functions as a training aid, the ManTIS training mode had to be designed using short and concise sentences, as it was intended to be viewed online using a computer screen. The amount of text needed to be minimal and pictures were used to replace text whenever necessary. A summary of the topic and a direct link to the ManTIS reference mode was required upon completion of each topic.

Task and user analyses were conducted to draw the design requirements. Since the ManTIS reference mode and training mode aimed to support a generic task, i.e. to test the base station, an informal task analysis method was employed. The purpose was twofold: (1) to understand the associative relationship between the properties of the topic and hence (2) to decide the access structure to the information domain. The results suggested four access routes conceptually adopted from the properties of each topic, such as technology, product description, etc.

User analysis recommended three levels of job function: engineers, technicians and experienced operators. However, they only represented two groups of expertise. Engineers and technicians were considered experts, whereas inexperienced operators were considered novice.

The requirements specification for the ManTIS reference mode and the training mode are listed in Table 23 and Table 24 respectively.

Table 23 – Requirements specification for ManTIS reference mode

Introduction	
Application	ManTIS Reference mode
Purpose	Online reference library for product and test procedures
Scope	Systems test area
Content	*Horizonmacro* product, GSM technology, test equipment and test procedures
Responsibility	Test System Engineering (TSE)

Task characteristic	
Type	Information retrieval (data sheet, schematic diagrams) to support the task
Main task	[1]- Initialisation and alarm test and [2]- Radio Frequency (RF) test
Description	Functionality and security tests

User characteristic	
Job level	Engineers, technicians (including Maintenance Online / MOL technicians)
Level of expertise	Experienced
Computing level	Frequent Internet (WWW) users

Requirements	
Source of content	Current documents: MPS, 3M
Format	HTML linked to PDF
Access	Web browser ActiveX control [toolbar, address bar omitted] on test station
Multimedia	Text with supporting pictures, technical data, schematic diagrams
Delivery	Intranet (online)
Retrieval type	Searching, indexing, browsing
Access mode	Non-sequential
Security	Read only
Entry points	Technology, Product, Test Equipment and Test Procedure
Direct link	Bookmarks on associated sections from other ManTIS modes
Training	Not required

Table 24 – Requirements specification for ManTIS training mode

Introduction	
Application	ManTIS Training mode
Purpose	Training
Scope	Systems test area
Content	*Horizonmacro* product, GSM technology, test equipment and test procedures
Responsibility	Test System Engineering (TSE)

Task characteristics	
Type	Learning about technical information related to the task
Main task	[1]- Initialisation and alarm test and [2]- Radio Frequency (RF) test
Description	Functionality and security tests

User characteristics	
Job level	Operators
Level of expertise	Experienced
Computing level	Frequent Internet (WWW) users

Requirements	
Source of content	ManTIS Reference mode
Format	HTML
Access	Web browser ActiveX control [toolbar, address bar omitted] on test station
Multimedia	Picture, animations and supporting text
Delivery	Intranet (online)
Retrieval	Browsing
Access mode	Sequential with *guided tours*
Security	Read only
Entry points	Technology, Product, Test Equipment and Test Procedure
Direct link	To ManTIS reference mode
Training	Not required

6.2.2 Step 2: Detailed design

Similar to the design of the ManTIS simulation mode, the strategy adopted during the iterative detailed design was to apply a formal method for the information and navigation design, and then to incorporate design heuristics during user interface design.

6.2.2.1 E-R design

As mentioned earlier, a topic was used to structure the content of both the ManTIS reference mode and training mode. A *topic* could be a base station, a test station or a particular technology. Those entities would need to relate to one another to compose a more meaningful and useful piece of information. For example, testing a base station called *Horizonmacro* required some piece of information about the base station, the test station or the technology. The relationship among entities is represented using an E-R diagram, shown in Figure 45.

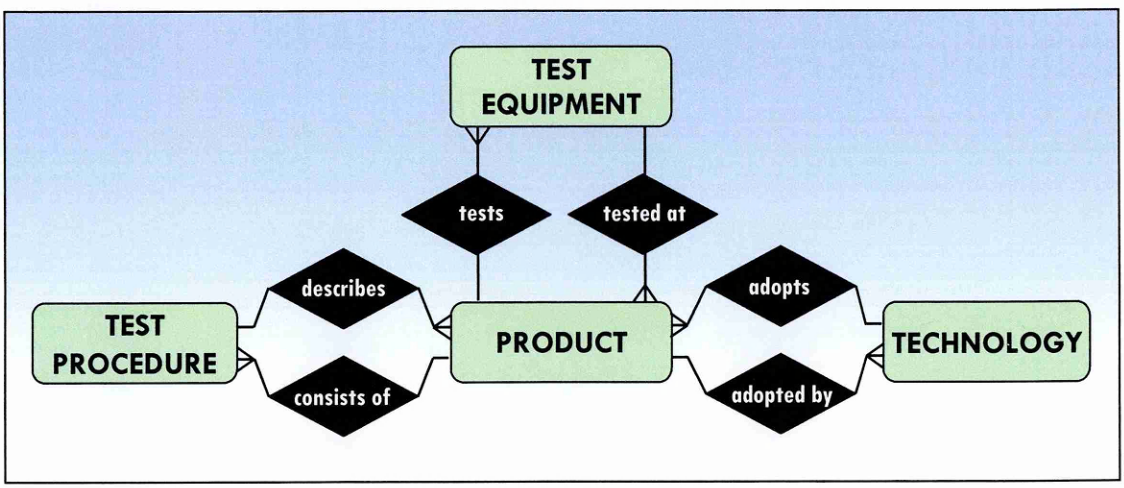


Figure 45 – E-R diagram of ManTIS reference mode and training mode

6.2.2.2 Slice design

Each topic (e.g. technology) had a number of attributes (e.g. GSM or 3G). These attributes were pieces of information (or *slices*) that would be grouped with other slices from other topics to form a new topic. Figure 46 shows the slice diagram for a *topic* called *Horizonmacro*.

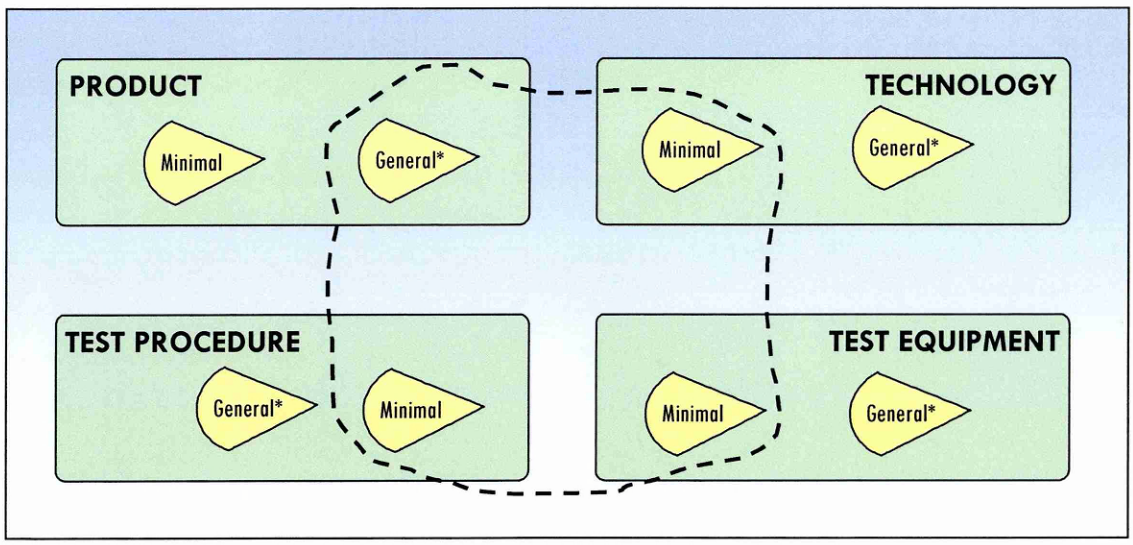


Figure 46 – Slice diagram of ManTIS reference mode and training mode

6.2.2.3 Navigation design

It was quite clear at this stage that a *topic* could be represented by relationships of entities, or a group of pieces of information from entities, which were also known as slices. The slices could therefore be used as entry points into the information domain. This also suggested the access structure of the ManTIS reference mode and training mode using a navigation tree. Each topic might contain one or more sub-topics, depending on the complexity of the topic. For example, the *Horizonmacro* topic was divided into technology, product, test equipment and test procedure.

Figure 47 shows the navigation diagram of the ManTIS reference mode and training mode. To a large extent, the diagrams are very similar. The only difference is that in the ManTIS training mode, the type of access between *product* and *technology*; *product* and *test equipment*; and *product* and *test procedure*, is *index guided tour* (indicated by dashed line), whereas in the ManTIS reference mode they use *index* only (indicated by solid lines).

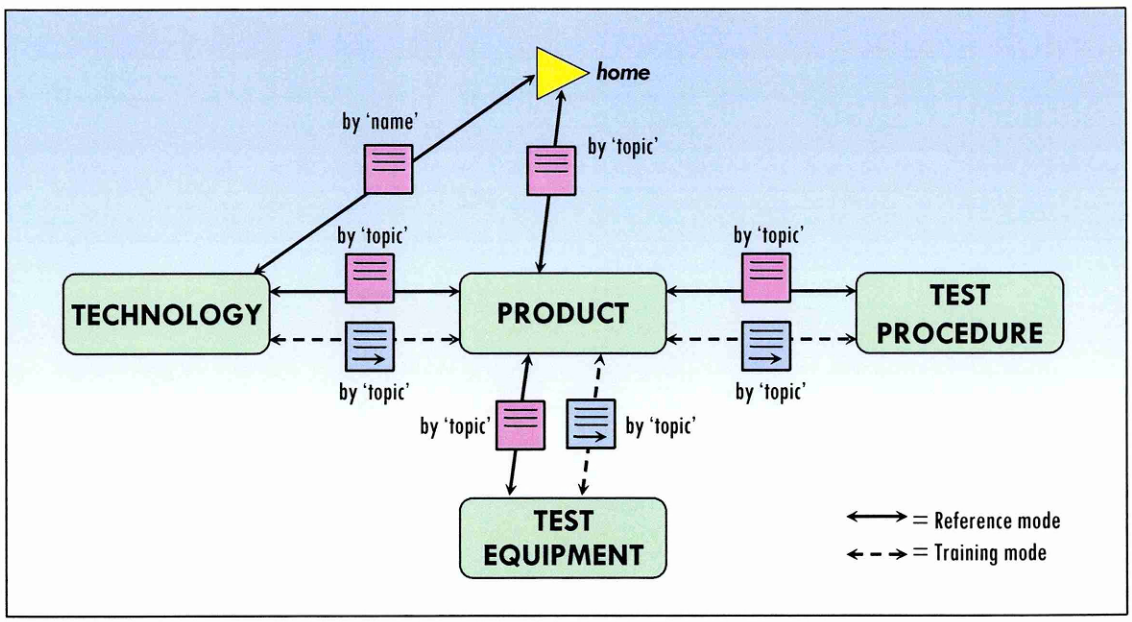


Figure 47 – Navigation diagram of ManTIS reference mode and training mode

6.2.2.4 User interface design

Table 21 and Table 26 present a summary of the design heuristics incorporated during the user interface design of ManTIS reference mode and training mode.

Like the ManTIS simulation mode, the user interface of the ManTIS reference and training modes were also split into three frames (Figure 48):

- *Top frame* indicated to the user the mode that was currently running, i.e. reference mode or training mode.
- *Navigation frame* displayed the access structure and all the individual pages arranged in a structured way.
- *Content frame* displayed the page selected from the navigation frame. At the top of each page displayed, the name and the reference of the page were shown.

The four main access points to the reference materials were located in the navigation frame. Each content page was marked by a page reference, showing its location within the hierarchy of the topic (see *page reference* in Figure 48). Thus, a page named 'The Air Interface' belonged to the 'Technology' section, 'GSM' and 'BSS Serial Data' sub-sections.

Table 25 – Design heuristics and how they were incorporated in ManTIS reference mode

Design heuristics	*Applied*	*Description*
1. Improve orientation		
Retrace mechanism	Yes	Breadcrumb metaphor (called page reference in Figure 48) helps indicate what page the users are currently viewing
Tree structure	Yes	On the tree menu, page being viewed is underlined (Figure 48)
Book/tab metaphor	No	-
2. Facilitate navigation		
Guided tours	No	-
Back/forward	No	-
Open new window	No	-
3. Node construction		
Chunked information	No	As a reference library, no need for displaying information in one screen
Filtering/suppressing detail (view detail)	No	-
Illustration, animation or video clips	Partial	Mostly textual description with technical data, schematic diagrams
4. User interface design		
Fixed screen size	Yes	Opens in full screen
Fixed position of objects	No	More flexible layout, as link to PDF files are required (Figure 49)
Split screen (frame)	Yes	Three frames are used
Judicious use of colour	Partial	Considered only when selecting background and foreground colour (dark over light)
Sans serif font family (e.g. Arial, Tahoma)	Yes	-
5. Improve coherence		
Tool tips	Yes	'On mouse over' effects are used
Typed link labels	Yes	All links are underlined
Overview diagram (site map)	Yes	-

Table 26 – Design heuristics and how they were incorporated in ManTIS training mode

Design heuristics	*Applied*	*Description*
1. Improve orientation		
Retrace mechanism	Yes	Breadcrumb metaphor helps indicate what page the users are currently viewing
Tree structure	Yes	On the tree menu, page being viewed is underlined
Book/tab metaphor	No	-
2. Facilitate navigation		
Guided tours	Yes	Navigation buttons (Figure 50) are provided to enable the user to move forward and backward according to a predefined route
Back/forward	Yes	
Open new window	Yes	Direct link to reference mode is opened in a new window (Figure 51)
3. Node construction		
Chunked information	Yes	Information is clustered so as to fit into one window without vertical or horizontal scroll. Scroll bar has also been omitted.
Filtering/suppressing detail (view detail)	No	-
Illustration, animation or video clips	Yes	Still pictures are used to simplify textual description
4. User interface design		
Fixed screen size	Yes	No screen adjustment required (opens in full screen)
Fixed position of objects	Yes	Fixed position of text-picture pair (Figure 50)
Split screen (frame)	Yes	Three frames are used
Judicious use of colour	Partial	Considered only when selecting background and foreground colour (dark over light)
Sans serif font family (e.g. Arial, Tahoma)	Yes	-
5. Improve coherence		
Tool tips	Yes	'On mouse over' effects are used
Typed link labels	Yes	All links are underlined
Overview diagram (site map)	Yes	-

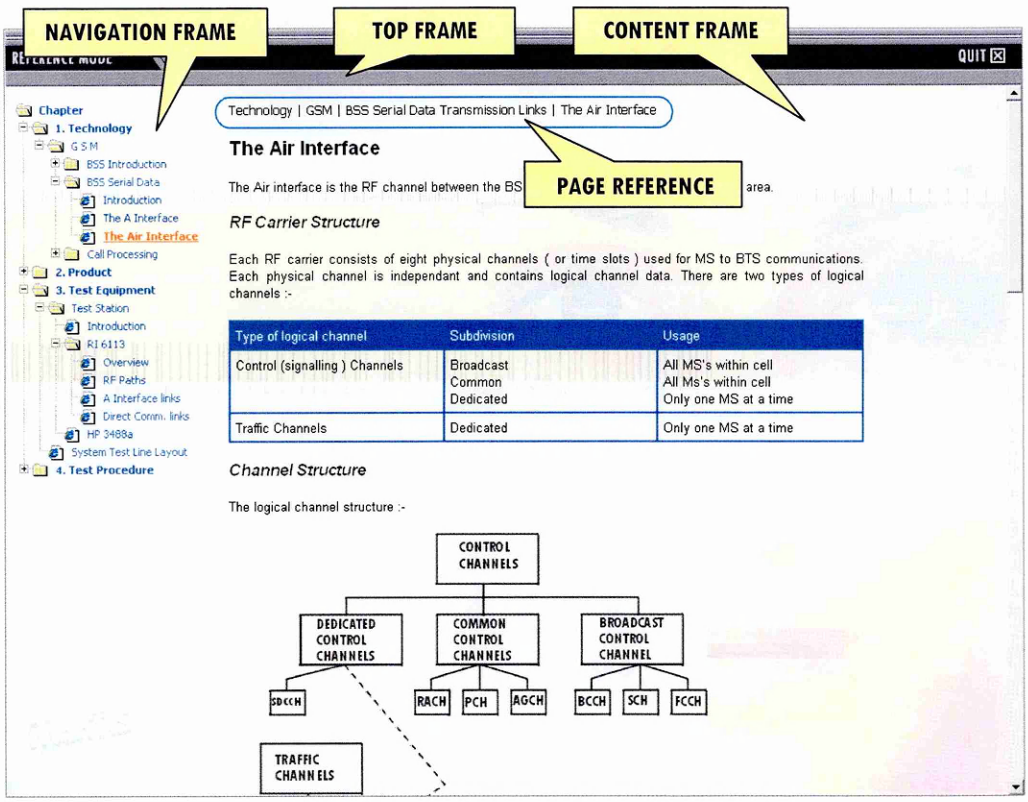


Figure 48 – User interface of ManTIS reference mode

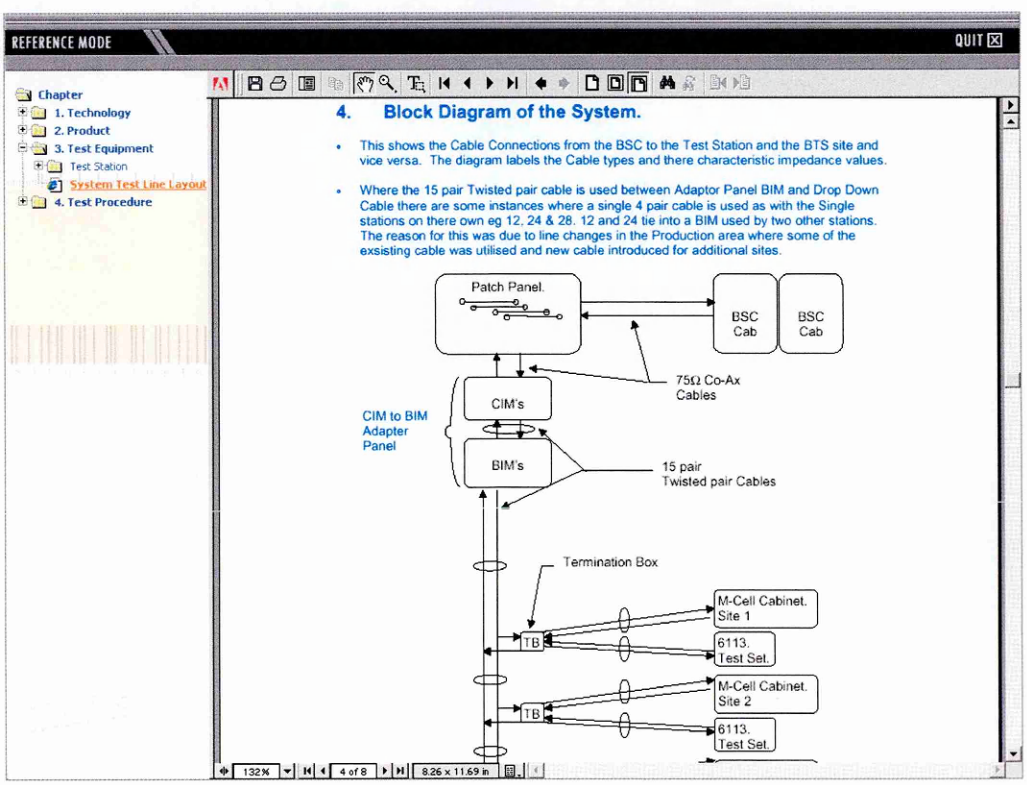


Figure 49 – Link to a PDF file

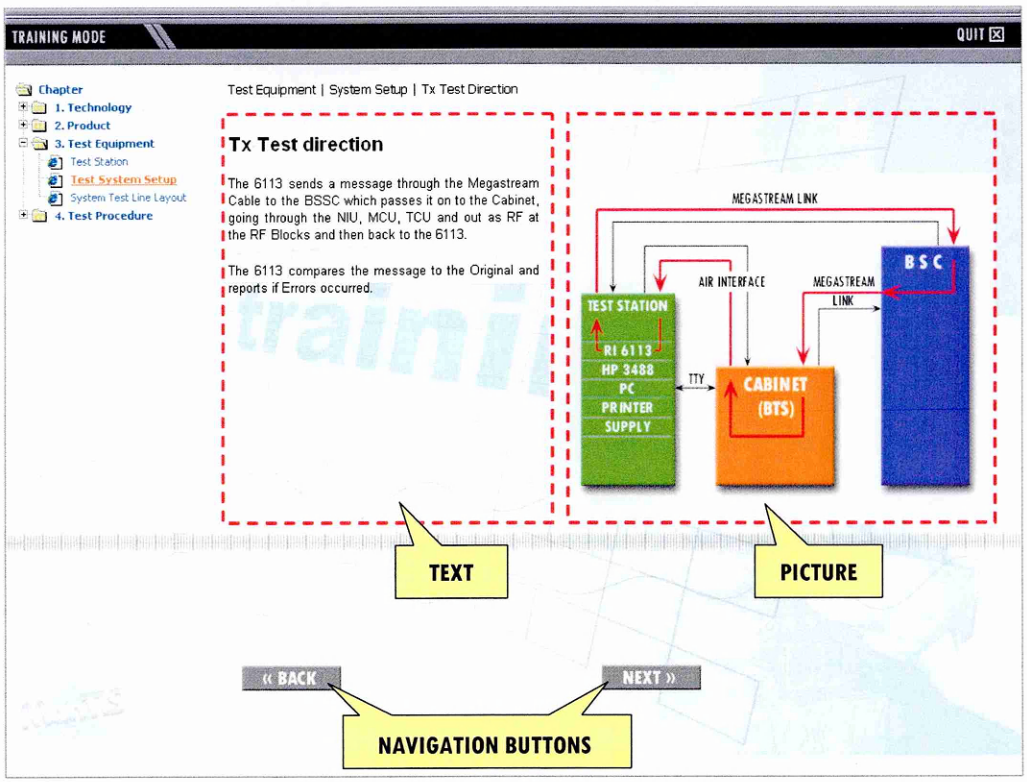


Figure 50 – User interface of ManTIS training mode

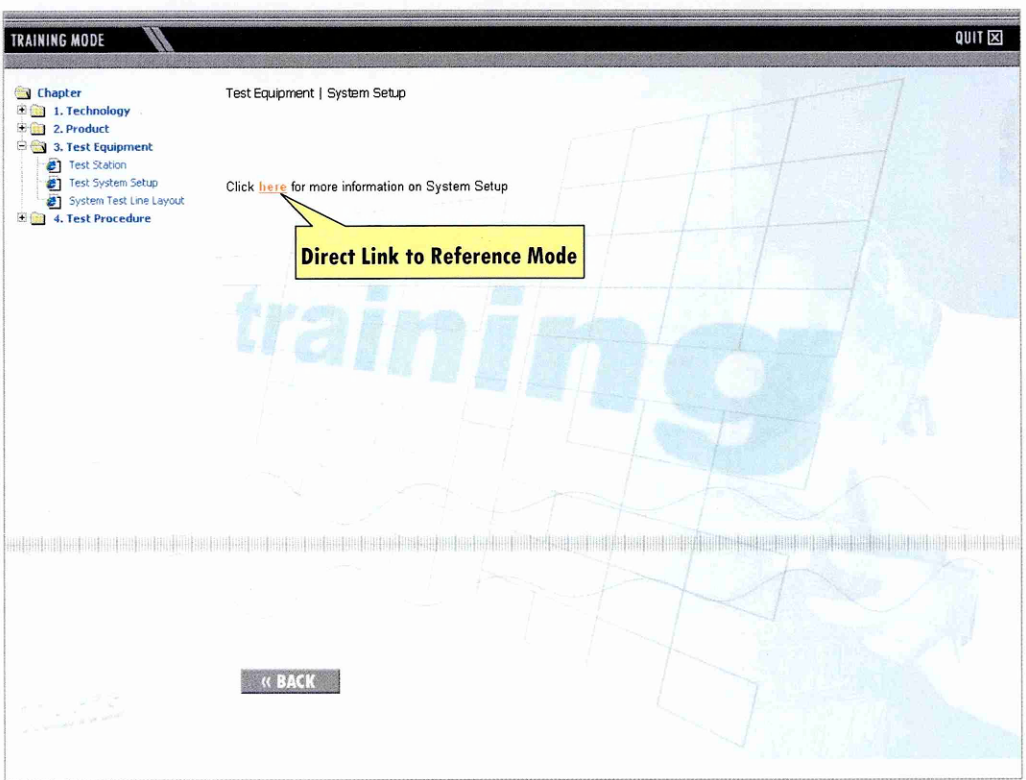


Figure 51 – Direct link to ManTIS reference Mode

The feature that distinguishes the ManTIS training mode from the ManTIS reference mode is the *forward* and *backward* navigation buttons that allows the users to follow a predefined route. To standardise the look and feel, the content part of the application was divided further into two parts: text on the left and picture on the right (Figure 50). The text and picture layout displayed on one page is believed to give a better description than either text or picture alone. Figure 51 shows a direct link from the ManTIS training mode to the reference mode. This allows the users to directly open a particular section in reference mode, for example the 'BSS Introduction', without having to navigate the unnecessary pages.

6.2.2.5 Design evaluation

Design evaluation was carried out through feedback from a number of technicians and engineers. The session primarily discussed the content of the reference mode and the training mode, rather than the user interface aspect, because the user interface was considered to be much simpler. During the design evaluation, issues relating to content creation and updates were raised by the technicians, as they had the responsibility for the system.

6.2.3 Step 3: Implementation

Both the ManTIS reference mode and the training mode were developed using HTML, allowing the content to be delivered through the company Intranet.

6.2.3.1 Conversion protocol design

Although their logical structures were defined using entity relationships, the ManTIS reference mode and training mode did not incorporate physical relational databases. Instead, *file system* and *file-naming* conventions were adopted to illustrate the relationship between a topic and its associated HTML pages. Therefore, the conversion protocol design was skipped.

6.2.3.2 Run time design

The information for the ManTIS reference mode was originated in various formats, produced using at least three different types of word processing packages. The document providers worked together with the ManTIS developer to identify the topic and to help structure the information. These documents were then converted into HTML. The layout and appearance were standardised to reflect the company's look and feel by using Cascading Style Sheet (CSS). With CSS, no manual formatting was required and this enabled document formatting on-the-fly, without a great deal of adjustment to the HTML document itself. Some JavaScript codes were employed to create the navigation tree (Windows Explorer-like navigation bar), where each topic could be expanded or contracted when necessary.

The ManTIS training mode was also developed using HTML, taking advantage of a number of reusable templates and codes from ManTIS reference mode. The majority of the information for the training mode was already in HTML. The layout and appearance were also standardised using the same CSS file that was used in the reference mode. In addition, the training mode also shared the JavaScript codes with the reference mode to create the navigation bar.

6.2.4 Step 4: Evaluation

Evaluations of the ManTIS reference mode and training mode were carried out after the system deployment, i.e. when the system had been in operation for approximately 4 months. Unlike the ManTIS simulation mode, no formal feedback session was conducted. This was due to the following reasons:

1. The user analysis suggested that the technicians and engineers were experienced users who understood the task very well

2. Compared to the operators, they also had more experience in using hypermedia or similar systems, such as the web

3. The user interface of both the ManTIS reference mode and the training mode were considered much simpler compared to the ManTIS simulation mode.

Therefore, during the evaluation, it was decided to incorporate a partial test of the success model. The objective was to investigate whether the ManTIS reference mode and the training mode had met the requirements, i.e. providing necessary information to support the systems test. According to the success model, this was done by analysing the degree to which the ManTIS reference mode and the training mode fitted the task.

The evaluation was conducted using a questionnaire. The questionnaire items used were the *Task-Technology Fit* (TTF) items listed Appendix B. The subjects were 19 experienced operators and technicians from 2 shifts, working in the testing area of the assembly line where the base stations were manufactured.

Once again, to simplify the questionnaire, 'reference mode' and 'training mode' were referred to as 'the program'. Each construct was measured with two (one positively and one negatively worded) questions, and each question was to be answered using a seven-point Likert scale, with 1 being 'strongly disagree' and 7 being 'strongly agree'. The questions were mixed, for example, a question measuring *currency* was followed by a question measuring *reliability*. Each subject was provided with instructions on how to complete the questionnaire and responses remained anonymous.

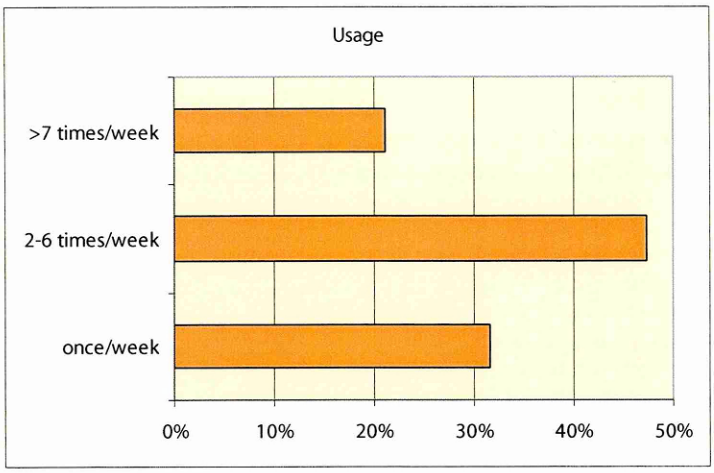


Figure 52 – Usage profile of ManTIS reference mode and training mode

Figure 52 provides details of the usage profile of the program. The graph indicates that the application was not used frequently. Only 20% of the subjects were using the application at least once per day, suggested that the information could be obtained elsewhere.

Table 27 – Users' perception of Reference and Training modes

Dimension	*Mean*	*SD*
Accessibility (ACC)	4.87	1.28
Currency (CUR)	3.76	1.54
Accuracy (ACU)	5.11	1.10
Meaning (MEA)	5.24	0.98
System reliability (REL)	4.47	1.29
TTF index	4.69	1.00

The results indicated that the program did not score equally well on each construct. In particular, the subjects felt that the content was not current enough (3.76 out of 7, with 7 being highly current). Moreover, their perception of *reliability* was close to neutral. Only *accuracy* and *meaning* had scores well above average (>4). Overall, the ManTIS reference mode and the training mode did not exhibit *task-technology fit* (TTF), as the index of 4.69 would indicate.

6.2.5 Step 5: Deployment and maintenance

To facilitate the maintenance of the ManTIS reference mode and training mode, a file structure was required to store HTML files on the web server. Figure 53 shows the file and folder structure that was intended to facilitate maintenance in the future.

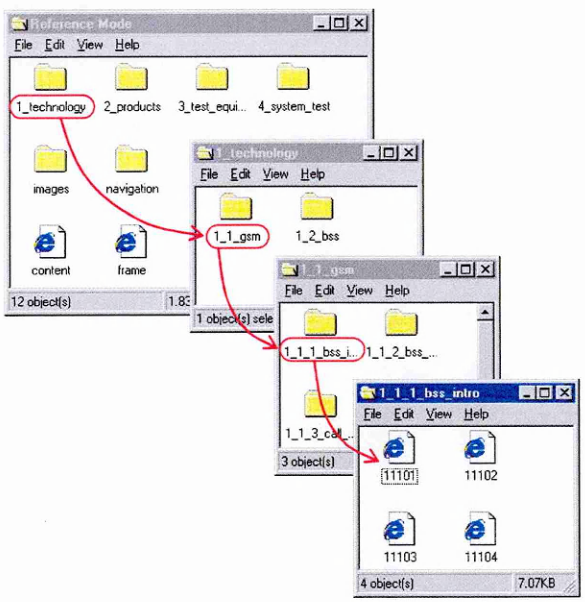


Figure 53 – File and folder structure of reference mode

A template was generated to assist the technicians in developing further topics in both reference mode and training mode. The template also allowed easy reproduction and duplication of the materials. To facilitate system maintenance, a *file name* convention was adopted to identify which file belonged to which topic. The key was the hierarchical structure of the files that composed a sequence of topics linked by forward and backward buttons. For example, a file called 11101.htm was the first page in a section called 'technology', sub-section 'GSM' and 'BSS Introduction' subsequently. If there was a subsequent page after this page, that page had to be named 11102.htm. At the end of the sequence, the page link to reference mode was constructed by the first 3 digits of the page.

The *file-naming* convention was also used in the ManTIS training mode to match the picture with the page to which it should be loaded. For example, a picture file called 11104.jpg belonged to the 11104.htm. The system would therefore identify whether a page consisted of text only or a combination of text and picture. If no picture files were available, by default the system would display a message to notify the users that the page contained text only. Both the folder structure and the file name convention would certainly help the system administrator to carry out updates and maintenance [Chan *et al*, 2000].

6.3 Case 3: Positrol help system

In 1998, a new set of quality system standards, known as TL 9000, was emerged as an extension to the ISO 9000 standard. TL 9000 obliges telecommunication industries to demonstrate their conformance in supplying high quality products to the customer. They are therefore required to successfully complete a third party registration audit from an accredited TL 9000 registrar.

Positrol, which stands for Positive Control, is part of the TL 9000 requirements that present an approach for controlling processes during the manufacturing of telecommunication products. Positive control means that the input variables of the process are controlled, instead of the product output variables. At Motorola, the input variables include the quality of the test stations and peripherals that are used to test the products. Positrol is, therefore, a preventive approach to conform to the quality standards by regularly inspecting the test stations and peripherals before they are used to test the products, in order to ensure that the test results are trustworthy.

A program to facilitate Positrol inspection tasks was developed at Motorola, covering the assembly line of the base station, including the systems test area. The program acts as an electronic checklist to log a set of Positrol tasks, carried out by a test operator at the start of a new shift. The program allows the data to be stored in a database, replacing a large number of paper checklists, kept for quality control purposes.

The Positrol tasks include inspection of the test station's components and connectors, which are used to test a base station. Upon completion of each task, an operator is required to select the appropriate button on the Positrol program to log the results. For experienced operators who regularly worked in the systems test area, this might be straightforward. However, operators from other areas are often required to work at the systems test area, and consequently they also need to carry out Positrol tasks. To assist them, an electronic help system, known as *Positrol help system* was developed. The system adopts a concept similar to the context-sensitive help system commonly found in any Windows-based program. The difference, however, is that the content of the Positrol help system is predominantly multimedia-based.

6.3.1 Step 1: Specification

When identifying specifications for the Positrol help system, it was imperative to consider that the end users were the shop floor operators. This meant that the information to support Positrol tasks had to be well structured, in order to reduce the learning time. A good design of hypermedia application should also encourage the end users to explore the content and find the information in the minimum time. To achieve this, users must be able to easily navigate the system and to retrieve the information they are searching for.

The problems with content currency in the ManTIS reference mode and training mode had led to a new requirement for the Positrol help system to take into account the capability of the system to allow manufacturing process experts, who were not hypermedia experts, to add and update the content of the Positrol help system according to the tasks to be supported, in addition to end users usability. This implied that the Positrol help system had to incorporate some kind of content management system, where the content could be generated on the fly and the hyperlinks could also be embedded automatically, to minimise maintenance effort.

Table 28 summarises the requirements for the Positrol help system.

Table 28 – Requirements specification for Positrol help system

Introduction	
Application	Positrol help system
Purpose	To provide more detailed Positrol inspection task description
Scope	Systems test area
Content	Systems test station
Responsibility	Test System Engineering (TSE)

Task characteristic	
Type	Task support
Main task	Test station and peripherals inspections
Description	Preventive maintenance

User characteristic	
Job level	Operators
Level of expertise	Novice
Computing level	Frequent Internet (WWW) users

Requirements	
Source of content	MPS 2602: Positrol procedure for systems test
Format	HTML
Access	Web browser ActiveX control
Multimedia	Text, still pictures, video clip
Delivery	Intranet (online)
Retrieval type	Browsing
Access mode	Non-sequential
Security	Read only
Entry points	Area
Training	Minimal

Features	
End user	Browse help content
Author	Post, update help content

6.3.2 Step 2: Detailed design

Both formal and informal design methods were employed during the iterative detailed design of the Positrol help system. A formal design method was required to create a structure for multimedia objects to which manufacturing process experts, in this case the technicians, could contribute the content of the help system according to the Positrol task. This included the management of Positrol tasks and their associated multimedia objects, as well as the management of hyperlinks from the task to the multimedia objects.

6.3.2.1 Entity-Relationship design

Within the Positrol help system, relationships exist between the Positrol task and a set of multimedia objects. In addition, a set of Positrol tasks would be used in a particular area within the assembly line of the base station. Therefore, three entities could be identified as *task*, *mm_set* (collection of multimedia objects) and *area*.

The relationship between *task* and *mm_set* is *one-to-one*, because one Positrol task can only have one set of Positrol help multimedia objects. However, one Positrol task from one particular area can also be used in other areas, and grouped as another set of Positrol tasks. This means that one area can have more than one Positrol task and one Positrol task can exist in more than one area. Therefore, the relationship between Positrol *task* and *area* is *many-to-many*. Figure 54 shows the E-R diagram of Positrol help system.

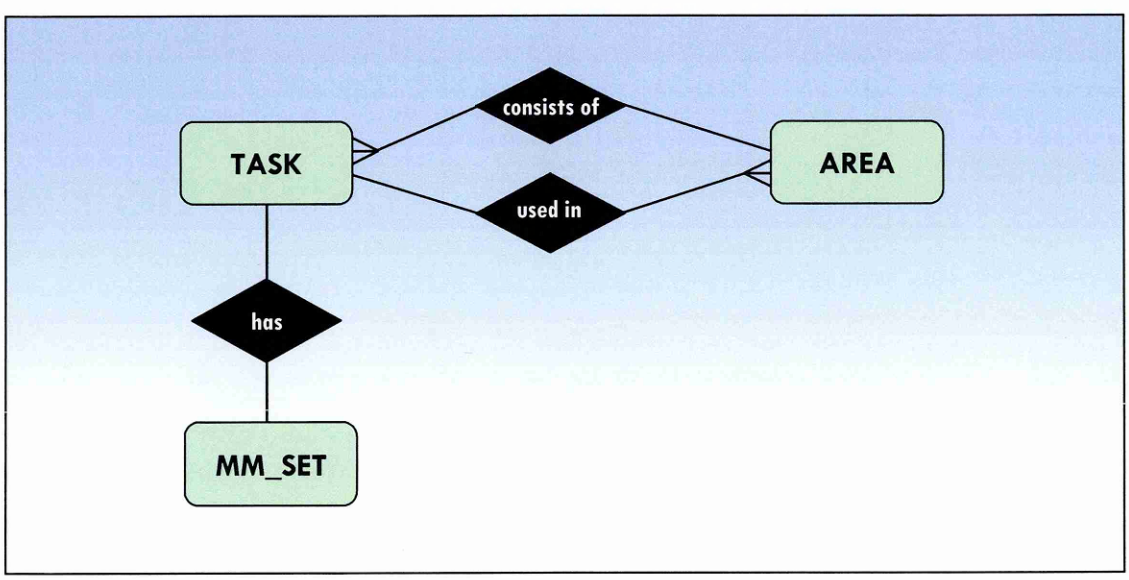


Figure 54 – E-R diagram of Positrol Help System

6.3.2.2 Slice design

In slice design, entity attributes were grouped in order to display the Positrol help system through the slice diagram. Since all attributes of the *mm_set* would be displayed, no slice diagram was produced.

6.3.2.3 Navigation design

Figure 55 shows the navigation diagram illustrating the access route to the Positrol help system. Upon execution of the main Positrol program, a list of areas where Positrol tasks need to be carried out is presented. The list is an index from the entity area. When the application was designed, the Positrol program and help system were deployed to Front End and Systems Test areas only. However, this was expanded into other areas within the assembly line.

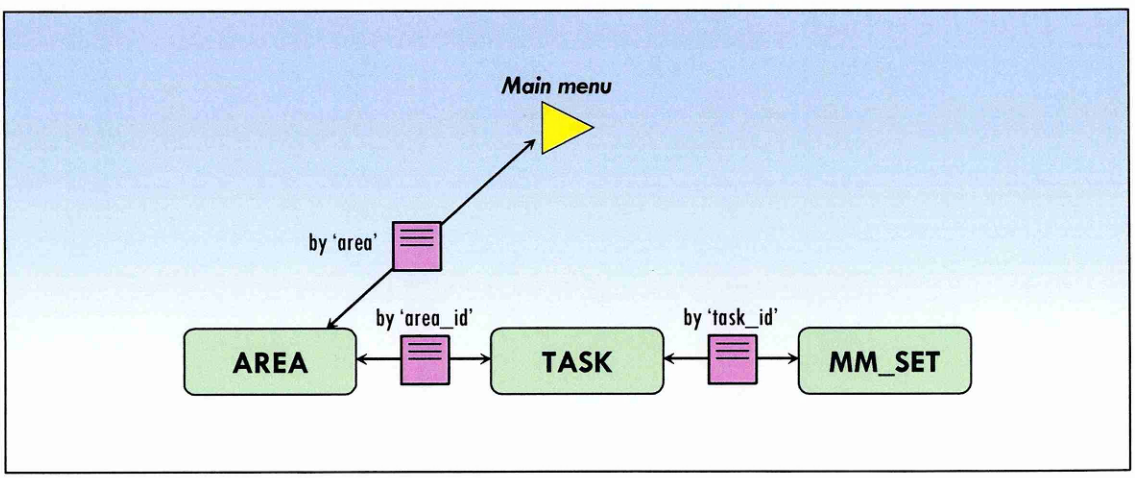


Figure 55 – Navigation diagram of Positrol help system

6.3.2.4 User interface design

The user interface of the Positrol help system was designed using design heuristics shown in Table 29, adopting the user interface of the ManTIS simulation mode. Perhaps the most important feature in the Positrol help system user interface was the fixed layout of multimedia objects. Not only did this help the users improve their orientation in finding the information, but it also reduced maintenance effort. With fixed position of multimedia objects, the layout could act as a template that could be reused to populate multimedia objects on the fly. Therefore, both automated page and hyperlink generation could also be achieved.

Table 29 – Design heuristics and how they were incorporated in Positrol help system

Design heuristics	*Applied*	*Description*
1. Improve orientation		
Retrace mechanism	No	-
Tree structure	Yes	Gives an overview of the area where Positrol task is applicable
Book/tab metaphor	No	-
2. Facilitate navigation		
Guided tours	Yes	Provided from within a task
Back/forward	No	-
Open new window	Yes	Like Windows help system, Positrol help system opens in a new window for ease of use
3. Node construction		
Chunked information	Yes	Positrol task was split into a number of sub-tasks, each associated with the help objects
Filtering/suppressing detail (view detail)	Yes	Help system is hidden by default
Illustration, animation or video clips	Yes	Each instance of Positrol task is represented text, still picture and video clip
4. User interface design		
Fixed screen size	Yes	480x480 pixels
Fixed position of objects	Yes	Layout of multimedia objects in Positrol help system is split into three areas to display textual description, two still pictures and one digital video clip. No vertical and horizontal scroll bars are provided
Split screen (frame)	No	-
Judicious use of colour	Yes	Two different colours were chosen to distinguish Positrol program and the help system
Sans serif font family (e.g. Arial, Tahoma)	Yes	-
5. Improve coherence		
Tool tips	Yes	'On mouse over' effects from Positrol program
Typed link labels	Yes	All links are underlined
Overview diagram (site map)	Yes	-

6.3.2.5 Design evaluation

A brainstorming session was attended by a number of test operators and technicians to verify the user interface design. More importantly, the session also discussed a new approach, also known as *collaborative authoring*, adopted during the system deployment, that allows technicians to contribute the content of the Positrol help system. The role of the hypermedia developer, on the other hand, has been limited to the creation of the Positrol help template, the database structure for multimedia objects and the access mechanism where the multimedia objects can be automatically linked from the Positrol task.

6.3.3 Step 3: Implementation

The Positrol help system was designed to operate in an online environment, accessible from any test station within the assembly line, using the Intranet. Therefore, HTML was chosen as the main enabling technology to implement the Positrol help system and to link each task in the Positrol program to its support information in the Positrol help system. Another reason for the selection was the availability of the HTML editing tool at the sponsor, with the capability of creating dynamic pages.

Although multimedia objects, particularly still pictures and digital video clips, were created and edited using similar methods to those used in the ManTIS simulation mode, they were not, at a later stage, prepared by the hypermedia developer. Instead, the technicians became involved in the development process, using the collaborative authoring environment.

6.3.3.1 Conversion protocol design

It was clear from the detailed design that, unlike the ManTIS simulation, training and reference modes, the multimedia objects in the Positrol help system could be managed using a relational database system. This was because the information in the Positrol help system was much easier to structure than that in the three ManTIS modes.

Conversion protocol design addresses the conversion of physical database tables, fields and keys in a relational database system, from logical E-R diagram produced during E-R design. Entities become tables and attributes become the table fields.

During the database design, the hypermedia developer was faced with the two options of either maintaining the multimedia objects using BLOBs (Binary Large Objects) fields, or using the combination of an RDBMS (Relational Database Management Systems) and a *file system*. The latter means that the database only stores the hyperlinks to the multimedia objects. The physical objects were in fact stored in a file system. In other words, the database only acted as a look-up table to locate where the each file was actually stored. Due to simplicity, and compatibility with the existing database system at the sponsor, the latter option (combination of RDBMS and file system) was adopted.

Microsoft Access was chosen for its compatibility with the SQL server to which it would be migrated in the future.

Indices in Figure 55 are represented as lists of query results. Queries were designed using SQL (Structured Query Language), a standard expression used to query information from a relational database system. Within the Positrol help system, a query served two purposes: (1) to display the index ('by area') in the navigation diagram (Figure 55), and (2) to populate the multimedia objects into the template according to the Positrol task that needed further explanation. Once again, Microsoft Access provided built-in function to design the queries.

Since the relationship between Positrol *task* and *area* was *many-to-many*, a linking table was required to establish a logical relationship between them. The linking table also provided a way to associate records from table task with table area, and helped ensure that there was no problem adding, deleting and modifying related data. More importantly, having a linking table would make the database structure more logical, to avoid the presence of multi-part fields, multi-valued fields and duplicate fields [Hernandez and Viescas, 2000]. The linking table was named *task_area*, and is shown in Figure 56. In this case, table *task_area* was also known as *transaction table* and linked a number of master tables, for example table task and table area.

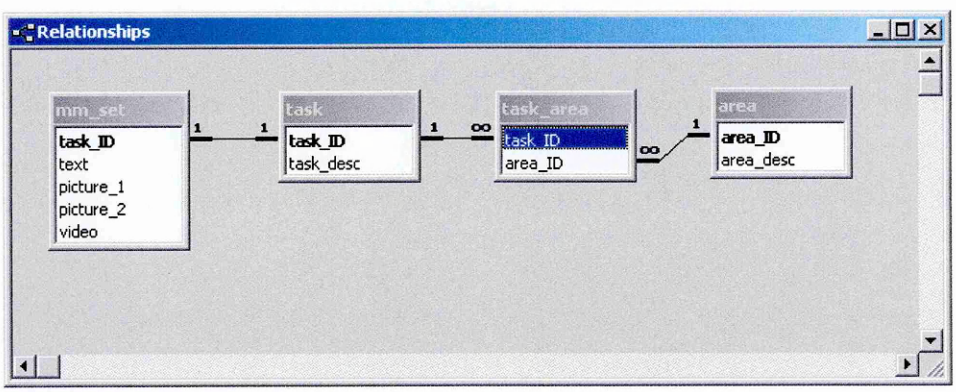


Figure 56 – Relationships between tables

6.3.3.2 Runtime design

Runtime design was a stage where the outputs from previous stages were actually programmed and integrated into the final system. Within the Positrol help system development, runtime design constituted the generation of the dynamic HTML pages using HTML template and automatic link generation, involving extensive interaction between the application and the database.

The template was a combination of the dynamic HTML page and the picture template created using image editing software, for example Adobe Photoshop. This was a one-off

job carried out by the hypermedia developer. The template was first linked to the database using ODBC (Open Database Connectivity). Every time a user clicked on a Positrol task through a hyperlink, that user passed the query parameter to the web server. The web server then processed the query parameter and executed the query to retrieve the correct file name and location of the multimedia objects stored on the server. Once the file name and location had been identified, the multimedia files were then inserted in the template and displayed to the user.

Automatic link generation was created using a relative link embedded from the Positrol program to the template and from the template to the multimedia objects. The Positrol program was linkable to the help system by utilising a query parameter embedded within the hyperlink. Upon execution, a link was automatically generated, depending on which task currently needed support.

The implementation of the Positrol help system utilised a web development tool called Macromedia DreamWeaver UltraDev. This software was used because it supports the latest dynamic web technologies such as ASP (Active Server Pages) and Cold Fusion. DreamWeaver UltraDev can be classified as a computer-aided software engineering (CASE) tool, for its capability to generate commonly used programming codes, for example to connect the page with the database through ODBC connection.

The use of dynamically generated pages would dramatically reduce the number of files to be maintained on the web server. With this approach, only one page was required in order to display a set of multimedia objects. This page was reused to display other sets of multimedia objects.

Figure 57 shows the screenshot of the Positrol help system. Each instruction on the Positrol task corresponded to a set of support media consisting of a digital video clip, at least one picture and text, displayed in a new window, as shown in Figure 57. The link between the task and these media was embedded using hyperlink. Since the support system was embedded within the task, support documents were no longer required during the execution of the Positrol task.

As stated in the design requirements, the Positrol help system had to enable the technicians to interact directly with the web server to publish the content. They needed access control and authorisation, which was previously held by an IT administrator. Therefore, a set of administration pages, for example, login page, upload page and database maintenance pages, were developed. This allowed the technicians to access the web server and database and, consequently, to put multimedia files in the appropriate location. Within the Positrol help system, this was known as the *collaborative authoring environment* [Tjahjono and Greenough, 2002]. Figure 58 shows one of the administration pages within the collaborative authoring environment that was used to upload the files to the web server.

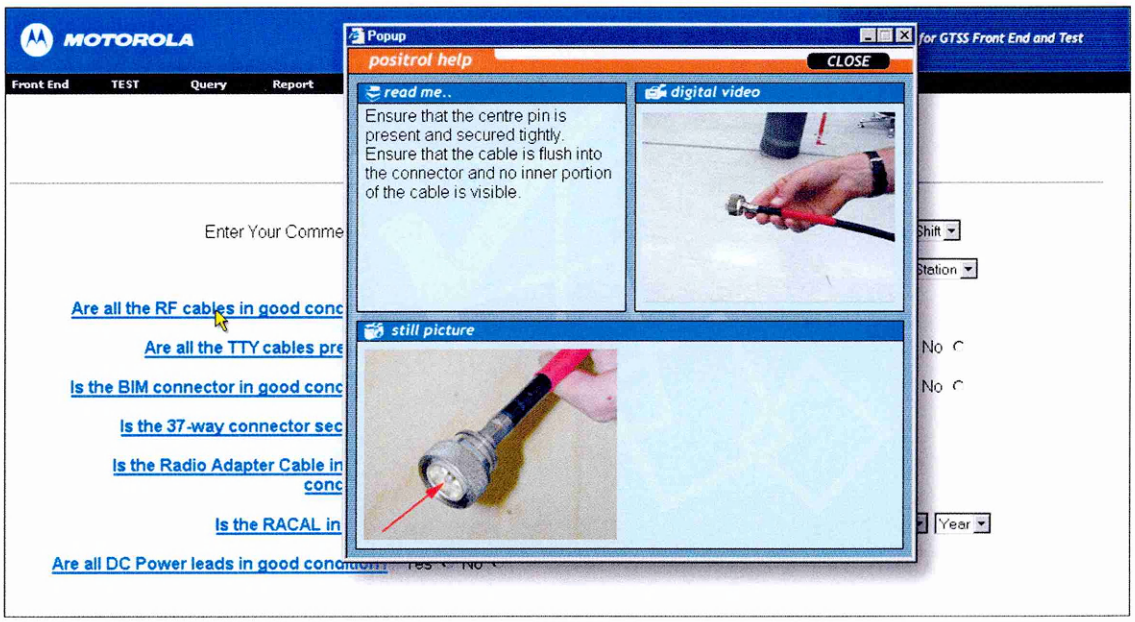


Figure 57 – Positrol help system

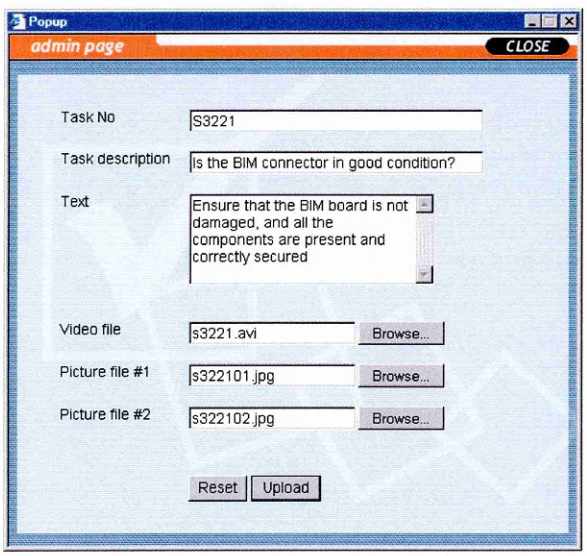


Figure 58 – Administration page

6.3.3.3 Content creation and population

Prior to database population, the multimedia objects needed to be collected and created, if they did not already exist. Technicians carried out task analysis and user analysis in order to understand the extent to which the multimedia objects would help the operators understand the Positrol tasks. The process continued with the selection of the media to be used to explain the task. A short, concise textual description about the task was provided, in addition to still pictures and digital video clips.

The technician then took still pictures, usually digital photographs, and edited them if necessary, so that they illustrated the textual description. A usability test was carried out to decide whether a short text and image alone were sufficient, or whether a digital video clip was required. In fact, some of the tasks did not necessarily need to be explained using digital video. During the usability testing, the technicians investigated usability problems that might occur, and continually modified the multimedia objects as required.

Having collected the multimedia objects, the next stage was to populate the database with the file name and folder location of the multimedia objects on the web server. This was carried out using an administration page, allowing the technicians to upload multimedia objects to the server and edit the relevant dynamic links on the database.

The administration page was also designed using ASP. This page required authorisation, so that the technicians were only able to access those folders on the web server and records on the database for which they were responsible. After uploading the multimedia objects to the web server, the technician filled in a form to create a record on the database referring to those multimedia objects.

6.3.4 Step 4: Evaluation

Evaluation of the Positrol help system was carried out using a mainly quantitative approach, where two sets of questionnaires were developed, one for the operators and the other for the technicians. The latter was to investigate the maintainability of the Positrol help system developed using the collaborative authoring environment. In addition, a feedback session involving a number of technicians was conducted, to obtain their perception of the usability of the collaborative authoring environment used to maintain the content of the Positrol help system.

6.3.4.1 Usability evaluation for operators

The first evaluation of the Positrol help system involved 16 test operators, using questionnaires. Eleven of them were experienced test operator and the remaining were novice test operators, represented by operators from other areas within the assembly line.

The questionnaire consisted of 10 items that were slightly reworded from the original items in Chapter 5, although they measured the same constructs, i.e. *perceived ease of use*, *perceived usefulness*, *task-technology fit* and *intention to use*. Each question was to be answered using a 5-point Likert scale, where 1 meant 'No' and 5 meant 'Yes'. The reason for using 5-point scale was that the operators were more comfortable with fewer options.

The questionnaire is shown in Appendix B.

The indices of the four constructs collected by the questionnaire are shown in Table 30. The subjects perceived the Positrol help system as *easy to use* and *useful,* indicated by the high mean values of both *perceived ease of use* (EOU) and *perceived usefulness* (U). They also rated the *perceived TTF* as high. More importantly, the *intention to use* for the subjects was also high, indicating the acceptance of the system.

Table 30 – Users' perception of Positrol help system

	Mean	SD
Perceived ease of use (EOU)	4.67	0.56
Perceived usefulness (U)	4.56	0.82
Perceived task-technology fit (TTF)	4.72	0.81
Intention to use (IU)	4.75	0.44

Further analysis was carried out to investigate two issues: (1) whether *perceived ease of use* (EOU) and *perceived usefulness* (U) affect the user's *intention to use* the Positrol help system; and (2) whether perceived *task-technology fit* (TTF) affected *perceived ease of use* (EOU) and *perceived usefulness* (U).

To do this, the mean values of *perceived ease of use* (EOU) and *perceived usefulness* (U) were correlated against the mean values of *intention to use* (IU). Similarly, the mean values of task-technology fit (TTF) were also correlated against the mean values of *perceived ease of use* (EOU) and *perceived usefulness* (U). The results are illustrated in Figure 59.

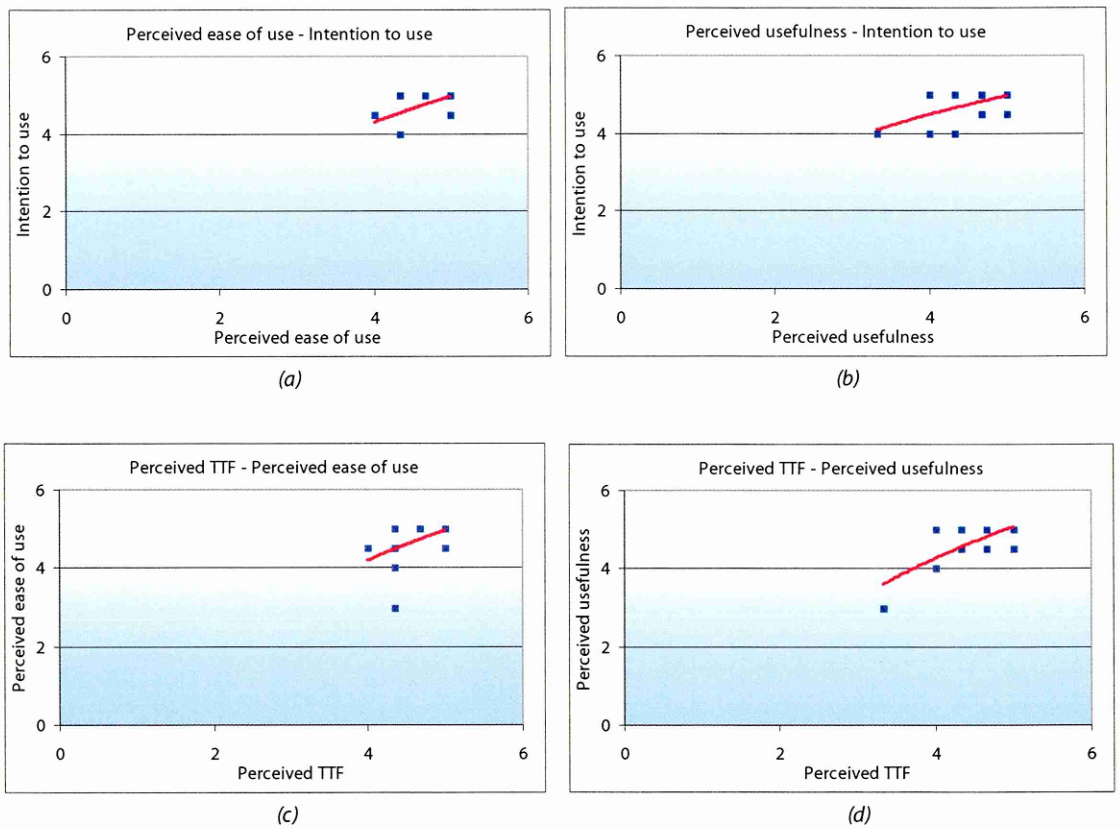


Figure 59 – Correlation between constructs in Positrol help system

From the above figure, it can be seen that the results confirm the success model. Both *perceived ease of use* (EOU) and *perceived usefulness* (U) have a positive correlation on *intention to use* (IU) as demonstrated in Figure 59 (*a*) and (*b*). Similarly, *task-technology fit* (TTF) was demonstrated to have a positive correlation to both *perceived ease of use* (EOU) and *perceived usefulness* (U), although a comparison between Figure 59 (*c*) and (*d*) suggests that the correlation of *task-technology fit* (TTF) to *perceived usefulness* (U) is slightly more significant than to *perceived ease of use* (EOU), as indicated by the gradient of the curve.

6.3.4.2 Usability evaluation for technicians

A secondary usability evaluation was conducted, in which a number of technicians were involved as the subjects of the study. They represented manufacturing process experts who created, updated and maintained the content of the Positrol help system. The evaluation aimed to investigate whether they could create the content of the Positrol help system and update it using the collaborative authoring environment. The usability of the collaborative authoring environment would therefore affect the maintainability of the content, which in turn ensured that the content was accurate, current, accessible or, in other words, fitted the task it supported (TTF).

The evaluation was conducted in two stages: a feedback session using a semi-structured interview and a questionnaire. The former was conducted prior to the deployment of the Positrol help system on the shop floor, whereas the latter was conducted after deployment.

The feedback session was attended by 6 technicians from two work shifts. The session started with a short demonstration on how the collaborative authoring environment worked. The environment was essentially a number of administration pages that allowed the technicians to upload the multimedia content to the web server, to manage the content according to the task and to update the content. In addition, a guideline, including a set of templates to create multimedia objects such as still pictures and digital video, was also required.

After a description about the entire process for developing the content of the Positrol help system, the subjects were asked their opinion about the collaborative authoring environment. This was crucial because, unlike the ManTIS, the Positrol help system would be jointly developed by the hypermedia developer and the technicians.

The results from the feedback sessions can be summarised as follows:

1. As with other procedural tasks, guidelines to create multimedia objects were provided. This was found useful since the technicians were not familiar with multimedia technology. The guidelines were also found efficient, particularly because they could reduce the learning effort by following a step by step instruction, for example on how to edit a video clip.

2. Once the media were created, the technicians could upload the files to the web server, using the administration pages. They were impressed with this, because previously the IT administrator had to modify the page in the server. The administration pages were found easy to use and user friendly.

3. However, an important feature that needed to be addressed was security and document control, including version control of multimedia objects.

A survey was conducted after the deployment of the Positrol help system. The subjects were, once again, the 6 technicians who had participated in the feedback session. The questionnaire aimed to obtain empirical data as to whether the collaborative authoring environment improved content maintainability.

The questionnaire for the technicians incorporated 5 dimensions of maintainability. Each of them comprised 2 items, giving a total of 10 items. The dimensions included *reusability*, *changeability*, *analysability*, *stability* and *testability*. As with the questionnaire for the operators, each question was also to be answered using a 5-point Likert scale, where 1 meant 'No' and 5 meant 'Yes'.

Appendix B lists the items chosen to measure maintainability of the Positrol help system, where *reusability*, *changeability*, *analysability*, *stability* and *testability* were coded as REU, CHA, ANA, STA and TES respectively.

The results of the questionnaire are summarised in Table 31. To obtain the index for each maintainability dimension, the mean values of each dimension were averaged.

Table 31 – Users' perception of maintainability

	Mean	SD
Reusability	4.58	0.51
Changeability	4.33	0.65
Analysability	3.17	0.72
Stability	4.08	0.52
Testability	4.25	0.45
MAINTAINABILITY index	4.08	0.78

The subjects considered the content to be easy to create, indicated by a high mean value of *reusability* (4.58 out of 5). This implies that the technicians could modify and create new content using both the template and the guidelines provided. The mean value for *changeability* was also high (4.33 out of 5), suggesting that they were able to update the content and the links. This result was not surprising, since no manual linking was required, because it was handled by an automatic link mechanism using *variable*, as opposed to *absolute,* address.

Unlike the first two dimensions, the mean value for *analysability* was 3.17 out of 5. This ambivalent decision was apparently due to the lack of version control of the multimedia objects. Although they were able to spot the wrong multimedia objects being displayed, they were nevertheless uncertain as to whether those objects were the correct version that had been authorised and approved by the configuration control.

On the other hand, the mean value for *stability* was moderately high (4.08 out of 5), indicating that the once multimedia objects were created there was no need for updates in plug-ins or drivers. The use of mature and *de facto* standard technology like HTML equally supported the stability of the system.

The majority of the subjects considered the system to be *easy to verify*, for example, after changing the text font and colour. Cascading Style Sheet (CSS) controlled the appearance of the system, ensuring that changes in this file affected all other files that shared the same CSS file. This was the reason why verification of the appearance was practically not needed. In addition, by using automatic link generation, no further

verification was needed, because as long as the database contained the accurate link information, the correct link would be embedded. This demanded that the technicians who were responsible for this had to carefully populate the content of the database.

Despite the moderately low mean value of *analysability*, overall, the use of the collaborative authoring approach resulted in easy maintainability of the Positrol help system, indicated by the maintainability index of 4.08 out of 5.

6.3.5 Step 5: Deployment and maintenance

In general, the deployment of the Positrol help system involved setting up the template files on the web server, setting up the database and the connectivity, publishing the administration page dynamic, and publishing the guidelines so that it was downloadable. Once deployed, the Positrol help system needs minimal maintenance, with the exception of the content itself. This, however, can be accomplished by the technicians (manufacturing process expert) through the use of the collaborative authoring environment.

Neither the hypermedia developer nor the IT administrator was involved in content maintenance. Not only did this release the hypermedia developer and/or the IT administrator from managing the files on the server, it also ensured the accuracy of the files to be uploaded by allowing the technicians to manage the content themselves. The quality of information in the Positrol help system became the sole responsibility of the manufacturing process experts. This made a more effective use of resources and contributed to agility, since they could reconfigure the documentation as soon as they became aware of the need for change. They did not need to wait for the hypermedia developer to do the work for them.

6.4 Case 4: Multimedia MPS

In order to deal with the problems associated with paper-based MPS, an electronic documentation system enhanced with multimedia objects was developed. The system is called *Multimedia MPS* (MM-MPS). The main purposes of the MM-MPS are to improve authoring process of the MPS; to increase management control of the documentation system; to speed up the approval process; to improve comprehension of current MPS and therefore to ensure the quality of task execution covered by the MPS.

To improve accessibility of the documents, MM-MPS paper-based MPS are delivered using web-based user interface through the company Intranet. With a web-based system, it was possible to develop a document management system that incorporated a workflow engine. This makes it easier for the Configuration Control Department to manage the MPS and to control the issue of the MPS numbers. In addition, it also allows the MPS authors to track the MPS during the sign-off process.

To ensure consistency of the MM-MPS, an authoring environment to write an MPS online is provided through the use of web forms. The forms act as a predefined template where the MPS authors can fill out the relevant information. MPS authors are not required to format the layout of the MPS since it is laid out automatically by the system. The web forms also impose on the author the requirement to write simple, concise sentences that comply with the online writing style.

The extensive use of multimedia objects to replace the amount of text in paper MPS makes it easier to understand the tasks by watching videos or a series of still pictures. This results in a more concise and easier to understand MPS.

The MM-MPS has the following features:

- It allows the end users to search and browse the MPS, using criteria such as MPS number, MPS title and MPS area, from a single web-based system, as opposed to multiple systems in the conventional system. Upon retrieval, brief information about the MPS was displayed on screen. Then the users can follow the hyperlink for full detail of the MPS, including the associated multimedia objects.

- It facilitates the authors to write the MPS online using electronic forms, to update the MPS with new issue control, to sign-off the MPS electronically and to track the status of the MPS during the sign-off cycle.

It enables administrator, possibly from the Configuration Control Department, to manage the MPSs and their associated meta-information, including the authors, and to take the ownership of the system.

6.4.1 Step 1: Specification

One of the requirements of MM-MPS was that it had to reflect the structure of the paper MPS, as shown in Figure 60. In general, the MPS consists of two parts: the procedure and the meta-information. The procedure is located in *section-6* of the MPS and is in fact the core of the MPS, describing the tasks. *Meta-information* includes everything else. Preliminary analysis suggested that an ideal solution would be to locate both *meta-information* and *section-6* in a common repository. However, there were a number of issues associated with that:

- It was not possible to incorporate *section-6* into the current database system at Motorola due to the nature of the procedures, which varied from one another. For instance, one procedure might have a flowchart and another might not. One might have two pictures and three videos, while others might have three pictures only.

- If the system restricted the way *section-6* was written, for example using HTML form, it would have made it even more complicated for the MPS authors to complete the MPS efficiently.

The above issues led to the decision to take *section-6* out of the database and leave the MPS authors to write *section-6* separately. Nevertheless, as in the Positrol help system, a template would be provided to assist the authors to type the relevant information and to link them to multimedia objects. The *section-6* files would be organised using a *file system* and directory structure. During retrieval of the MPS, the system would populate both meta-information from database and *section-6,* using an automatic hyperlink.

In addition to structure, the proposed MM-MPS should allow:

- The *users* (operators) to search and to browse only the released MPS.

- The *authors* to create new MPS, to continue editing new MPS and to update the released MPS.

- The *administrators* to edit user profile, to generate MPS number automatically and to sign-off MPS.

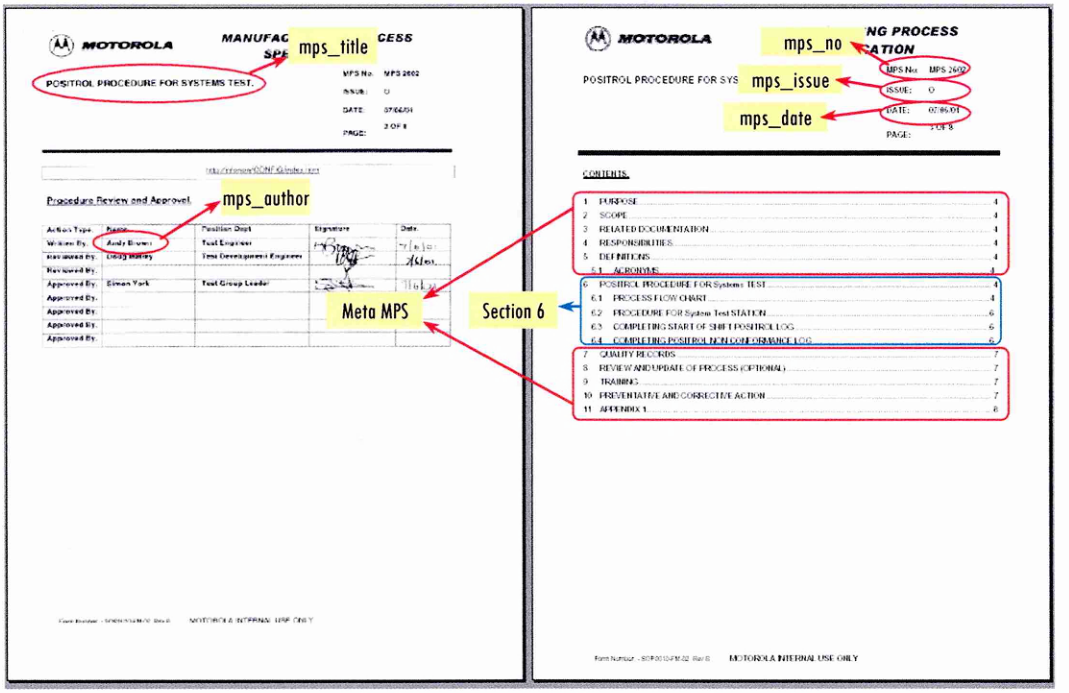


Figure 60 – Structure of MPS

Table 32 shows the requirements specification for MM-MPS.

Table 32 – Requirements specification for MM-MPS

Introduction	
Application	Multimedia MPS (MM-MPS)
Purpose	To provide more detailed Positrol inspection task description
Scope	Test Systems Engineering (TSE)
Content	Manufacturing Process Specification (MPS)
Responsibility	Test System Engineering (TSE)

Task characteristic	
Type	Task support, documentation system
Main task	Assembling and testing base stations
Description	-

User characteristic	
Job level	Engineers, technicians, operators
Level of expertise	All
Computing level	All

Requirements	
Source of content	Current paper MPS
Format	HTML
Access	Web browser ActiveX control
Multimedia	Text, still pictures, video clip
Delivery	Intranet (online)
Retrieval type	Searching, browsing
Access mode	Hybrid
Security	Read, write
Entry points	MPS number, MPS title
Training	Minimal

Features	
End user	Search and browse MPS. Detail is displayed using multimedia objects
Author	Write, update, sign-off MPS
Administrator	Manage MPS, author

6.4.2 Step 2: Detailed design

The detailed design of MM-MPS followed both formal and informal design methods. To a large extent, the formal method was adopted from the Positrol help system, where the PS author would create and populate multimedia object themselves, using a step by step guide or template created by the developer.

6.4.2.1 Entity-Relationship design

When viewing a multimedia MPS, the operators, as main users, will be interested in aspects such as the MPS itself, MPS author, MPS status, and the area where the MPS was used. By looking at the meta-content of the MPS, the users may also be interested in more details about the actual procedure. Based on these requirements, seven entities were identified. They were *Meta*, *Person*, *Area*, *Session*, *Status*, *Role* and *Decision*.

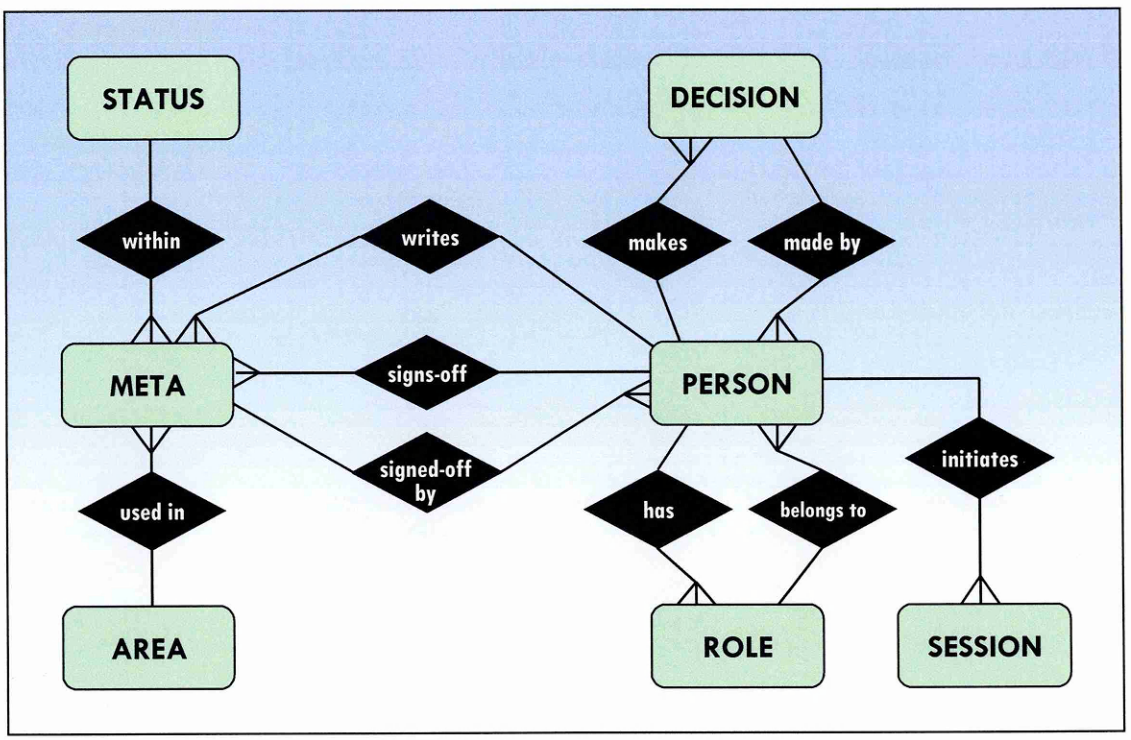


Figure 61 – E-R diagram of MM-MPS

Figure 61 shows the E-R diagram for MM-MPS. The relationship between *Status* and *Meta* is *one-to-many*. This is because a status can only be applicable to a particular MPS. In other words, no MPS can have more than one status. However, the relationship between *Person* and *Role* is *many-to-many*. This is because one person can have more than one role and one role could belong to more than one person. For instance, a person called *Joe* is a *reviewer* in MPS number 5012 and also an approver in MPS number 5013. Similarly, a role called reviewer may consist of *Joe*, *James* and *Jerry*.

6.4.2.2 Slice design

In this stage, entity attributes were grouped to display an MPS. Here, the entity *Meta* was split into four slices for a number of purposes, including to display an MPS to the end users and to display an MPS to authors for both authoring and tracking. For each entity, there were at least two slices: general and minimal. The general slice, which contained basic information such as *mps_no*, *mps_title*, *mps_status*, *mps_area*, *mps_no*, *mps_issue*, *mps_commerce_id* and *mps_date*, was the head slice.

Slices may be combined to construct a hybrid slice to represent how information from different entities can be organised into more meaningful chunks of information.

Figure 62 illustrates one of the slice diagrams produced in multimedia MPS.

In *meta* entity, its minimal slice contains two attributes: *mps_no* and *mps_title*, for example, '5012' and 'Quality Control for Systems Test', which for most purposes suffices to identify an MPS. The full slice contained the full information about the MPS, including *mps_keyword*, *mps_purpose*, *mps_scope*, *mps_reldocs_no*, *mps_reldocs_title*, *mps_def_acro*, *mps_responsibilities* and *mps_def_desc*. The actual procedure in that MPS is represented by a slice called *section-6*. Both full and *section-6* slices are required to display the MPS and are accessible from the general slice.

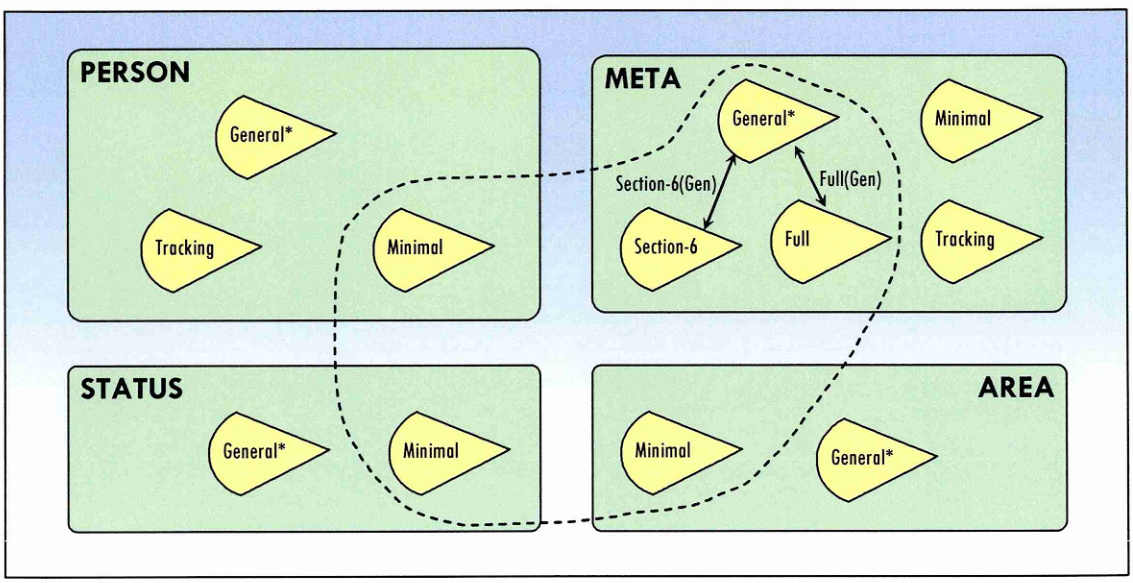


Figure 62 – Hybrid slice diagram to display an MPS

Table 33 lists the slices and their attributes to display an MPS. In conversion protocol design, hybrid slices are primarily used to construct queries in a relational database system.

Table 33 – Slices and their attributes to display an MPS

Entity	Slice	Attributes
Meta	General	mps_id, mps_no, mps_issue, mps_title, mps_status, mps_date
Meta	Section6	flowchart, picture_files, video_files
Person	Minimal	commerce_id, name_first, name_last
Status	Minimal	status_description
Area	Minimal	area_description

6.4.2.3 Navigation design

Figure 63 shows the main menu of the MM-MPS. There are four access points into the system, i.e. through user sub-menu, author sub-menu, administrator sub-menu and help sub-menu. User sub-menu is the only entry point to the end user to retrieve an MPS. Author sub-menu is the environment for the MPS authors to write an MPS online. In addition, authors will also be able to sign-off their MPSs when ready. Admin sub-menu is used to update the data on the database, such as details about an author. Both Author sub-menu and Admin sub-menu are password protected. The help sub-menu allows both end users and authors to find further assistance on how to use the system.

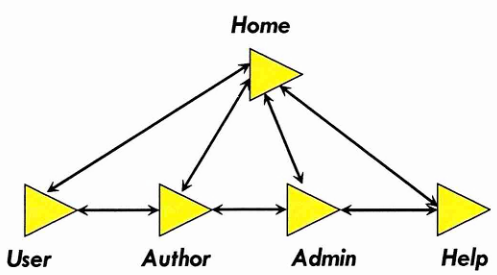


Figure 63 – Navigation diagram of MM-MPS (Main menu)

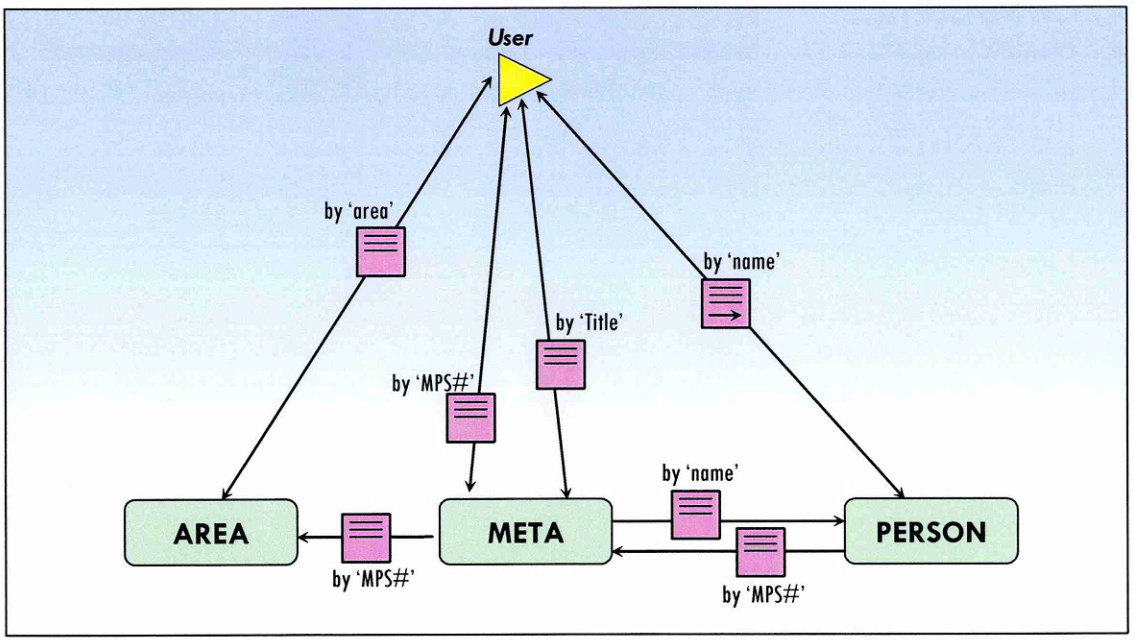


Figure 64 – Navigation diagram of MM-MPS (User sub-menu)

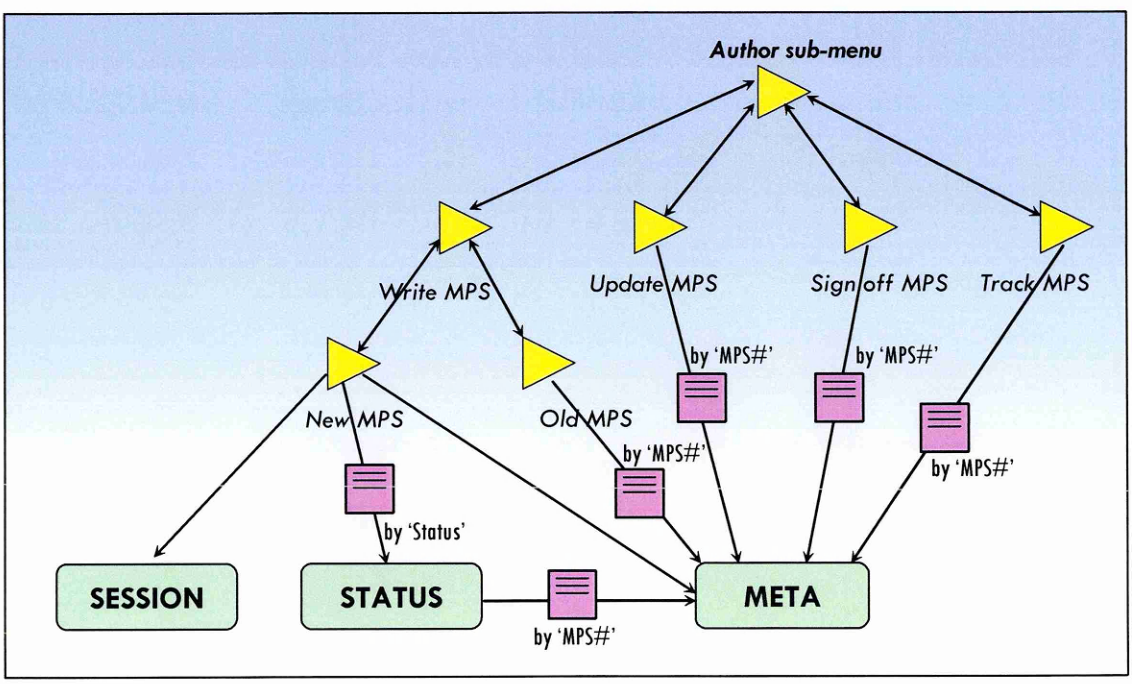


Figure 65 – Navigation diagram of MM-MPS (Author sub-menu)

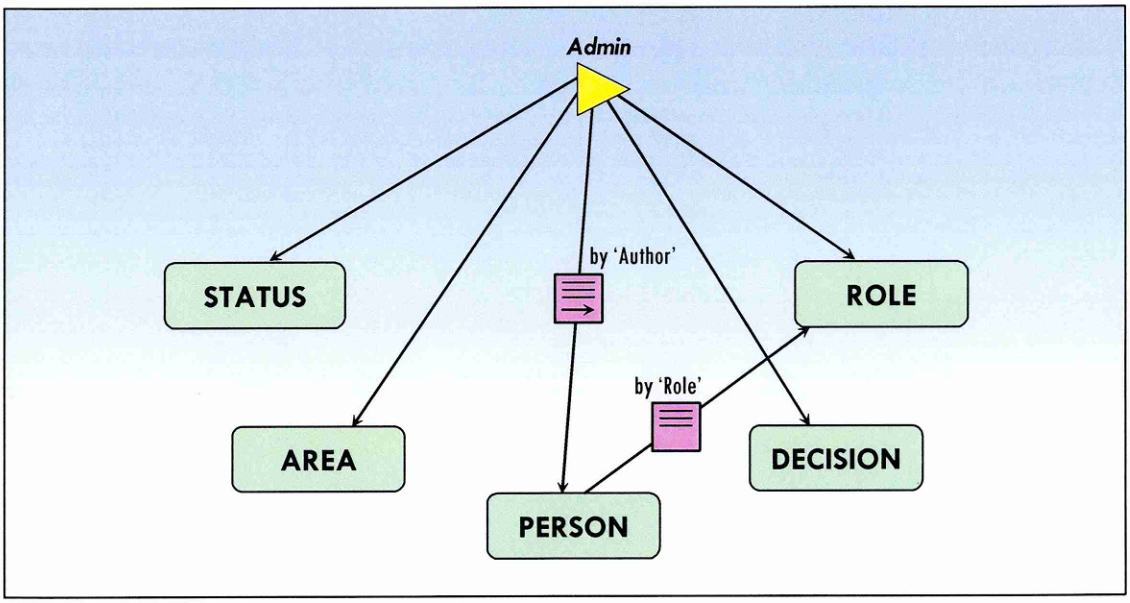


Figure 66 – Navigation diagram of MM-MPS (Administrator sub-menu)

6.4.2.4 User interface design

In this stage, each major window and the navigation path between windows were sketched, as shown in Figure 67. The user interface was designed using high-fidelity prototyping. Design guidelines followed the design heuristics shown in Table 34.

In addition to the user interface design for MM-MPS, this stage also designed the user interface for the MPS author and for the administrator. As specified in the design requirements, MM-MPS in fact provided a better collaborative authoring environment than that in the Positrol help system. In MM-MPS collaborative authoring, not only could MPS authors upload the files to the server, they could also write the MPS online, update the MPS and sign-off the MPS electronically. Therefore, the pages and the associated scripts had to be created.

Table 34 – Design heuristics and how they were incorporated in MM-MPS

Design heuristics	*Applied*	*Description*
1. Improve orientation		
Retrace mechanism	No	-
Tree structure	No	-
Book/tab metaphor	No	-
2. Facilitate navigation		
Guided tours	Yes	Entry point to the MPS from list of query result
Back/forward	No	-
Open new window	Yes	Detail of section 6 is provided in a new window
3. Node construction		
Chunked information	Yes	Main procedure was split into a number of sub-tasks
Filtering/suppressing detail (view detail)	Yes	Section 6 was intentionally hidden. 'Click for detail' button was provided
Illustration, animation or video clips	Yes	Procedures are represented with text, still pictures and video clips
4. User interface design		
Fixed screen size	Yes	Opens in full screen
Fixed position of objects	Yes	Content frame is fixed into two parts to display Meta information and the section 6 (actual procedure)
Split screen (frame)	Yes	Three frames were used
Judicious use of colour	Partial	Considered only when selecting background and foreground colour (dark over light)
Sans serif font family (e.g. Arial, Tahoma)	Yes	-
5. Improve coherence		
Tool tips	Yes	'On mouse over' effects were used
Typed link labels	Yes	All links are underlined
Overview diagram (site map)	Yes	-

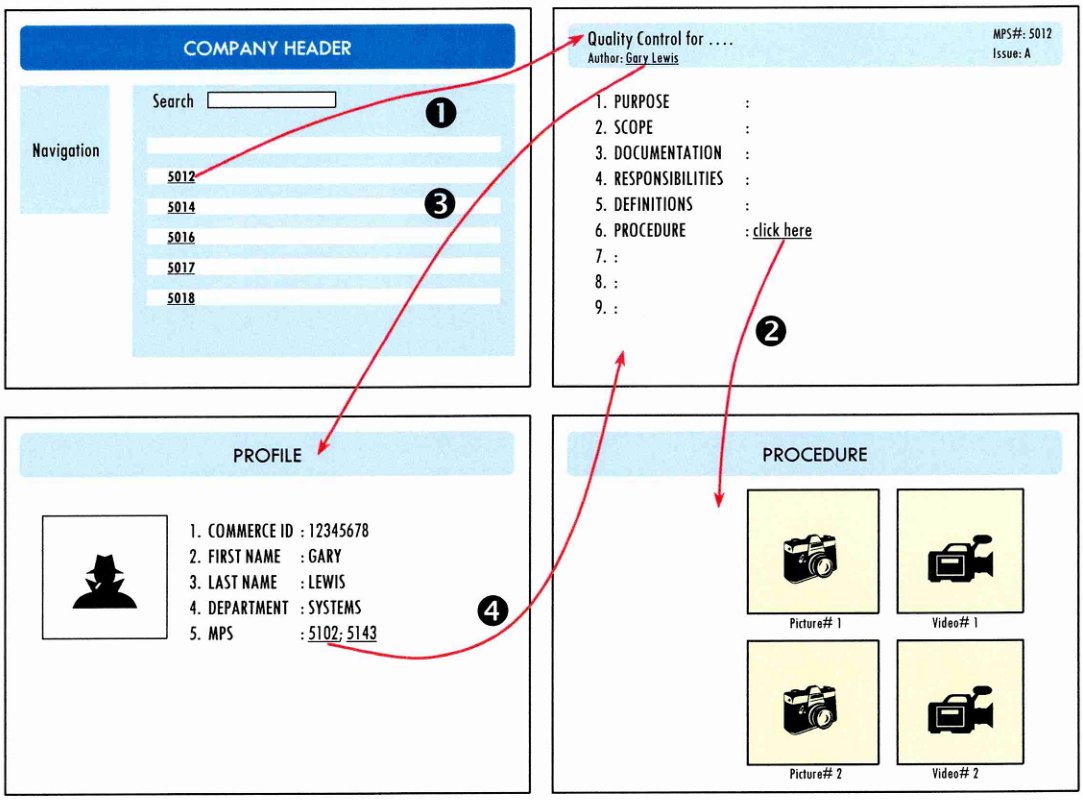


Figure 67 – User interface design and navigation path

6.4.2.5 Design evaluation

A brainstorming session was carried out during the design evaluation, to demonstrate the user interface design of MM-MPS. The session was attended by prospective users of MM-MPS, i.e. operators. It was explained that the MM-MPS utilised an electronic documentation system and adopted the same technology as the Positrol help system, in which multimedia objects, such as pictures and video clips, were to be used to reduce the amount of text in paper MPS.

A storyboard was used to explain the access route to view a particular MM-MPS. The operators were then asked about their expectations and for feedback as to whether they liked the new user interface format of MPS. The results from the brainstorming session were not surprising. Comparing MM-MPS with the traditional paper MPS, it was clear that they were quite happy with the overall concept of online document creation.

6.4.3 Step 3: Implementation

Since MM-MPS was to be delivered using the company's Intranet, a dynamic HTML was chosen as the enabling technology. Dynamic HTML uses a template where the content is populated from the database or, if not, from the *file system*.

6.4.3.1 Conversion protocol design

In this stage, the logical E-R design was converted into tables, columns and keys. Once again, Microsoft Access was chosen because of its ease of use and the compatibility with the MS SQL server.

A linking table was required in order to establish the relationship between *Person* and *Meta* in Figure 61. The linking table shown in Figure 68 was called *mm_sign*. The table *mm_sign* links two master tables *mm_meta* and *mm_user*.

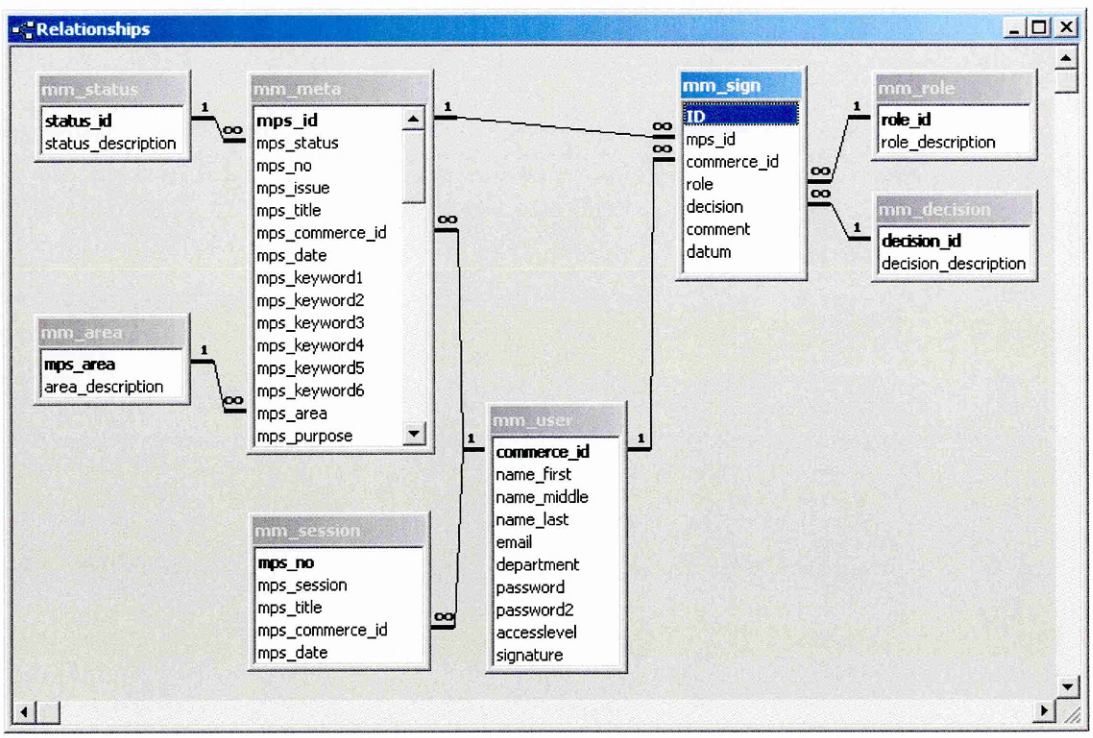


Figure 68 – Relationships between tables

The relationships in Figure 68 were also used to design two queries, namely *mm_qmeta* and *mm_qtrack*. The former embodies the hybrid slice in Figure 62, while the latter is used to display information used to track the MPS status during the sign-off process. Figure 69 shows the query design for *mm_qmeta*.

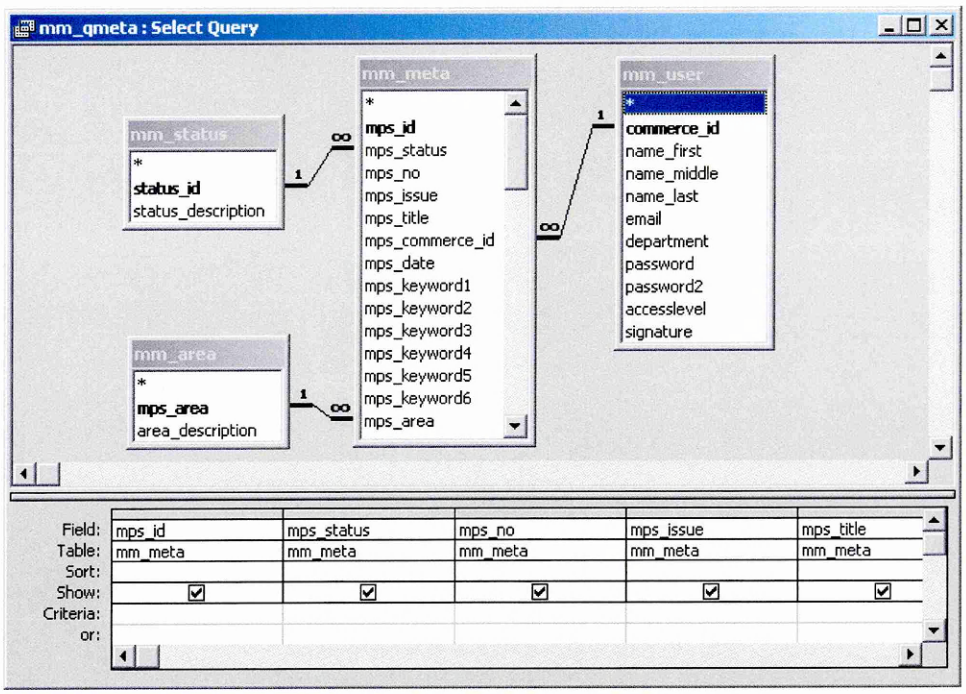


Figure 69 – Query design to display an MPS

6.4.3.2 Runtime design

This stage constituted generation of dynamic pages using Active Server Pages (ASP) that mostly involved interaction between the web server and the database.

As with the Positrol help system, Macromedia DreamWeaver UltraDev was used to implement MM-MPS, including the generation of a number of dynamic HTML pages using Active Server Pages (ASP). With ASP, only one page was required to display a particular MPS and this page would be reused for other MPS.

To incorporate section-6 into the main page, a button labelled 'retrieve section 6' was provided. When clicked, it would call an ASP server object to load the section-6 file from a predefined location on the web server.

As mentioned previously, the initial idea was to convert all the information from paper MPS into database tables. However, the major issue relating to a purely database driven MPS was that *section-6* appeared to be less structured, due to the nature of the procedure. Some might contain flowcharts and some might not. Some might use quite a few pictures, but some might only have a number of videos.

Because of the difficult of creating uniformity and standardisation in terms of both structure and layout, *section-6* was left outside the database system and was stored using a file system. Upon execution, the system would incorporate section-6 into the main page.

Figure 70 shows the screenshots of MM-MPS. To retrieve an MPS, a user must search (or browse) on the MPS number or MPS title. Upon retrieval, a list of the MPS that match the search criteria will be provided. The user can click the hyperlink for further details. The meta-information of that particular MPS is now displayed. Should the complete procedure be required, the user may proceed by clicking the hyperlink that will display section 6 (the actual procedure) in a new window.

In addition to the functionality for the user, a number of modules were developed to fulfil the design specification, particularly for authors and administrators. A large number of scripts were created on each module, particularly for the sign-off process.

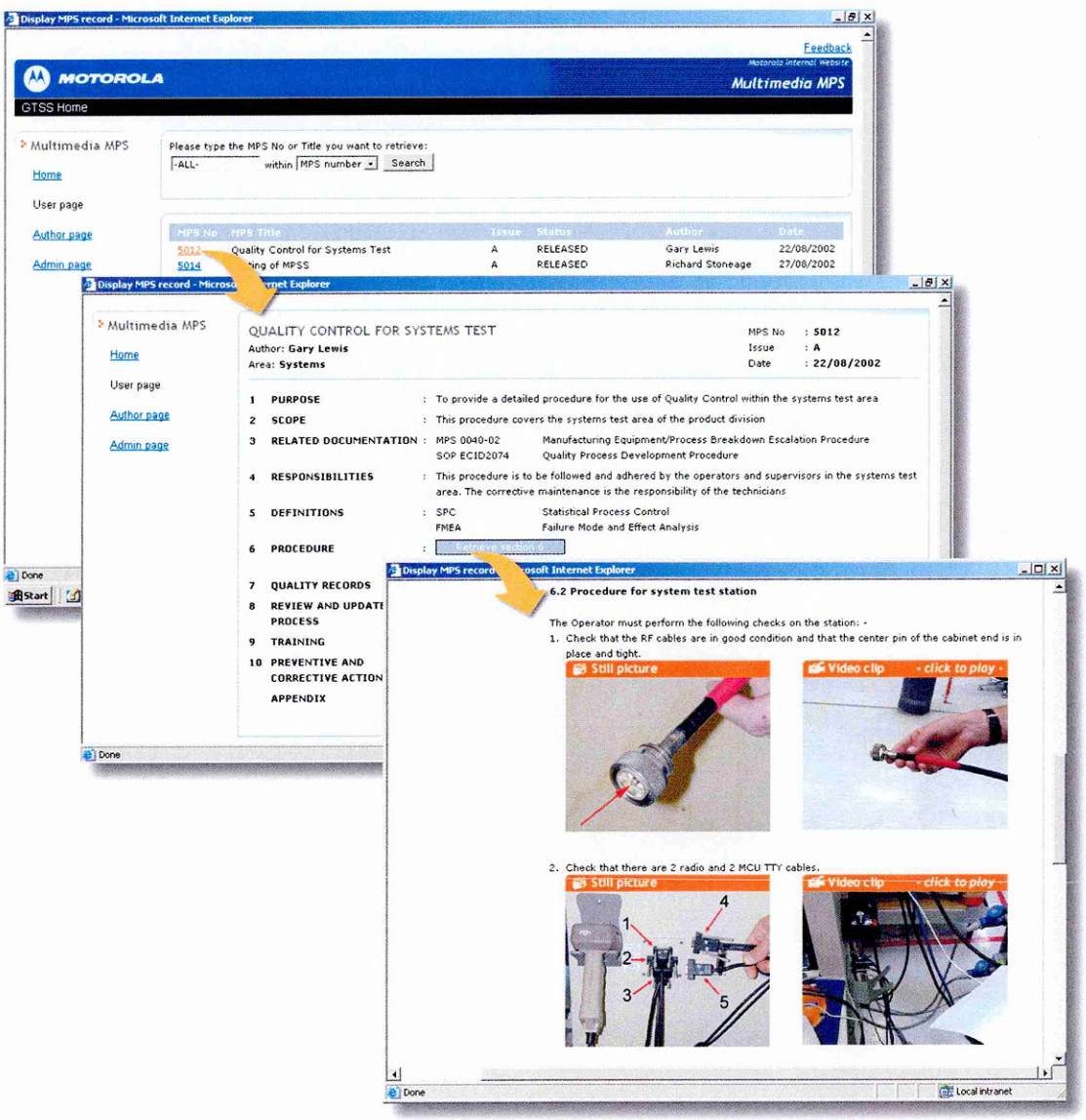


Figure 70 – Screenshots of MM-MPS

6.4.3.3 Content creation and population

As with the Positrol help system, the MPS author carried out task analysis to study how the textual description in paper MPS could be best supported by multimedia objects. The most appropriate media were then selected. They were then edited and collected for uploading to the server.

Having created multimedia objects to be used in the MPS, the MPS author used the collaborative authoring environment to write a new MPS or to update with the new issue (Figure 71). The environment provided a fixed template where the MPS author could fill in brief information about that particular MPS. Since the MM-MPS would be viewed directly on the computer screen, it was important to use the appropriate writing style, as well as the minimum amount of text.

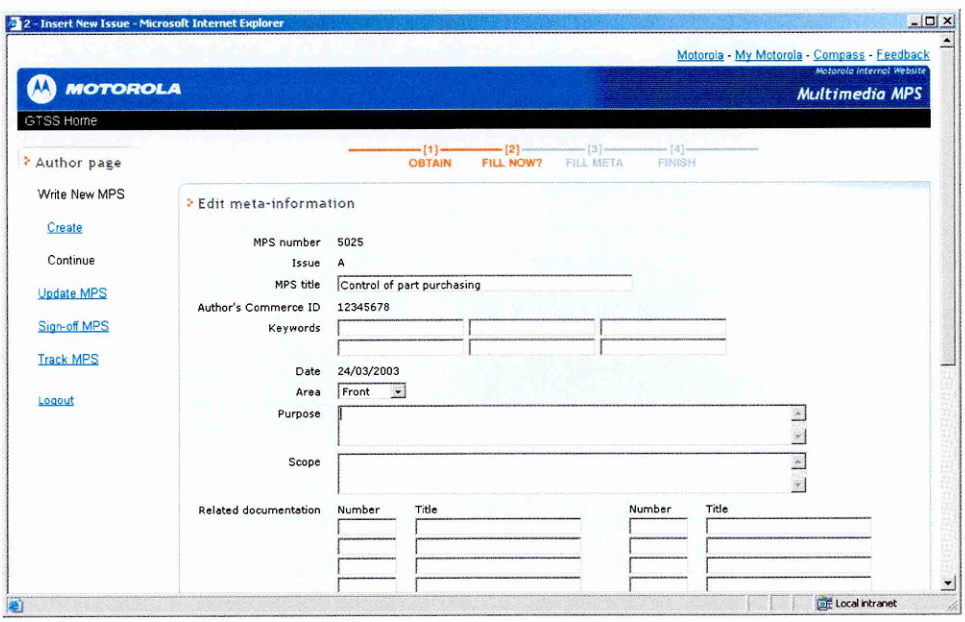


Figure 71 – Authoring a multimedia MPS

6.4.3.4 Authorisation

After authoring, multimedia objects were uploaded onto the web server and the detailed information was stored on the database. Once completed, the MPS status was set to *draft*, enabling the author to send the document to be signed off.

Unlike paper-based sign-off, electronic sign-off in MM-MPS was carried out through email. The MPS author first raised a sign-off request by nominating the reviewers and the approvers for that MPS (Figure 72). Upon submission of the sign-off request, the system sent emails to the nominated reviewers and approvers. The email contained a notification that a new MPS was waiting for sign-off. The reviewers/approvers were then instructed to follow a link embedded on the email.

The link directed the reviewers/approvers to join the collaborative authoring environment. The reviewers/approvers viewed the MPS to be signed-off, and they might either approve or reject the MPS (Figure 73). Upon rejection, the reviewers/approvers had to give their reasons for rejection. This would be logged onto the server and subsequent notification about the result of the sign-off would be sent back to the MPS author, once all reviewers/approvers had given the appropriate response.

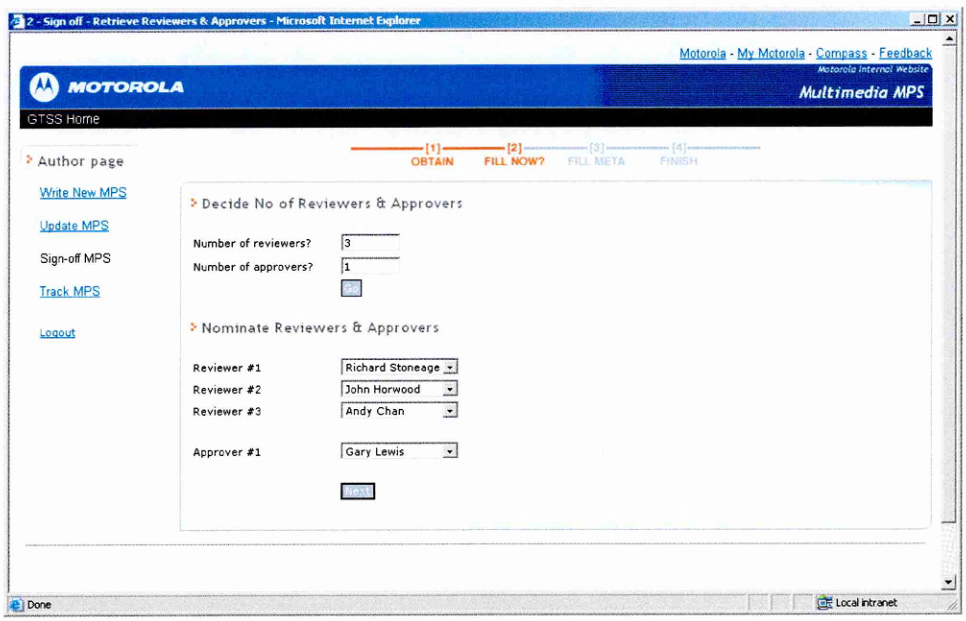


Figure 72 – Sign off multimedia MPS

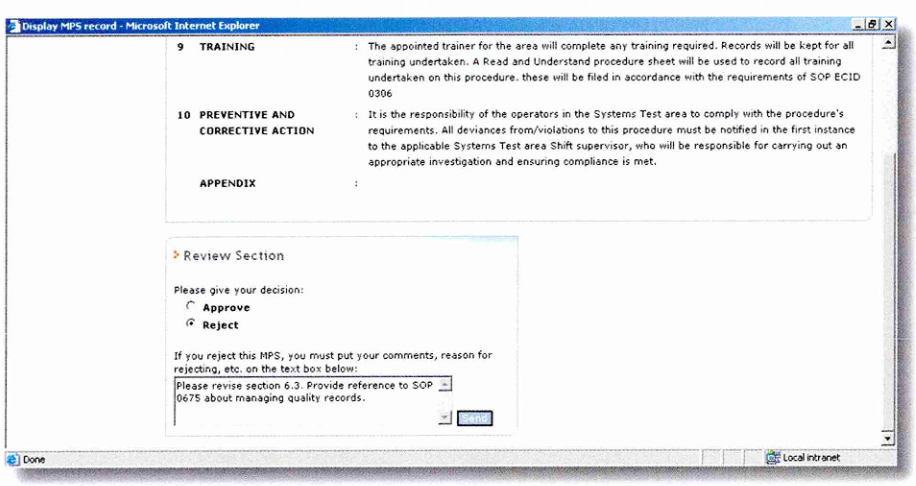


Figure 73 – Review section

6.4.4 Step 4: Evaluation

Evaluation of MM-MPS was carried out prior to system deployment, using a questionnaire followed by a brainstorming session. The evaluation of MM-MPS primarily aimed to investigate whether the MPS authors could use the collaborative authoring environment to create a multimedia MPS. Unlike the previous three cases, the end users, i.e. operators, were not involved during the evaluation. This was because the content and layout of MM-MPS had been made similar to that of the Positrol help system. The results from evaluation of the Positrol help system indicated that the users considered the system usable.

The questionnaire was given to 7 subjects (MPS authors and manufacturing process experts). The questionnaire incorporated the maintainability constructs as in the Positrol help system. However, the questions were slightly modified to include open-ended questions (Appendix B), allowing the subjects to give their opinion about the system.

The results from the evaluation can be summarised as follows:

1. ***Reusability***. The subjects agreed that MM-MPS was easy to create through reuse of user interface components and multimedia objects. The collaborative authoring supported this by providing a workflow of MPS creation, from obtaining the MPS number to editing the content. The step-by-step guides were useful to ensure that the records were filled properly.

2. ***Changeability***. Collaborative authoring also provided an easy to use interface to update the MPS. The system allowed the MPS author to retrieve the meta-information of the MPS to be updated, before updating the content. The old MPS would not be deleted; instead it would be archived once the new issue had been signed off. This allowed tracing and version control of the MPS to be applied.

3. ***Analysability***. The electronic sign-off in the collaborative authoring environment ensured that the content and multimedia objects used in MM-MPS were reviewed and approved before being published. The approval workflow also ensured that the MPS was released by the configuration control.

4. ***Stability***. The maturity of the underlying technology used in MM-MPS was apparently considered crucial by the MPS authors. They preferred to use one technology that was likely to survive for a long period of time. For that reason, they approved the HTML (or the web) as a medium for MM-MPS delivery.

5. ***Testability***. Collaborative authoring was found to be effective in addressing the problems associated with validating electronic documentation, such as the Positrol help system. The changes made on the MPS could be verified immediately. The updated MPS would have the same MPS number, but a

different issue number. The collaborative authoring environment automatically managed these updating and validating mechanisms.

6.4.5 Step 5: Deployment and maintenance

Unlike the previous three cases, the first deployment of MM-MPS had actually begun prior to the evaluation of the collaborative authoring environment carried out by the MPS author. The purpose was to verify the workflow of MPS generation and to compare it with the current process.

The deployment included the installation of the database and the collaborative authoring environment on the web server. In addition, registration of MPS authors was also done at this stage to provide them with adequate access privileges, for instance to write and to modify files on the server.

During this step, very little maintenance had been done by the hypermedia developer to MM-MPS. Once the format and the structure were verified, the MM-MPS would never be deleted from the system. Updates would add new records to the database, and consequently would append appropriate folders for the actual procedure (*section-6*). Maintenance activities for the MM-MPS were carried out by the corresponding MPS author, as in the traditional paper MPS development process.

6.5 Analysis

The evaluation of ManTIS simulation mode has indicated that the users' intention to use the application depends on the perception that the system is *useful* and *easy to use*. The correlations between both task-technology fit (TTF) and *perceived usefulness* and *perceived ease of use* imply that the hypermedia developer is required to fully understand the task to be supported.

In the case of ManTIS reference mode and training mode, the results from the system evaluation showed that the application was easily accessible from the workplace. ManTIS reference and training modes also showed advantages compared to paper-based processes, in terms user friendliness. However, there had been long delays in preparing the materials because the manufacturing process experts relied upon the hypermedia expert to transfer the materials for ManTIS reference mode and training mode. There was no means of enabling process experts to update the content without understanding HTML codes, resulting in poor currency of the content.

The above problem initiated a new requirement during the development of the Positrol help system, to allow the content providers, i.e. manufacturing process experts, to contribute the content of Positrol help system. Therefore, the development methodology needed to be modified to include the collaborative authoring environment, where the content creators could create and update the content of the Positrol help system without the involvement of the hypermedia developer.

During the evaluation of the Positrol help system, comments from users indicated that their understanding about Positrol tasks had improved through the use of the help system. The effort required to train new operators was also reduced, allowing supervisors and technicians to spend more time in other development projects.

However, from the evaluation that involved the technicians, it was apparent that the analysability of multimedia documents in the Positrol help system was not as high as the other dimensions in maintainability. This was due to the fact that the collaborative environment did not provide sufficient control over the multimedia files presented on the Positrol help system, as there was no approval process to ascertain that the multimedia objects presented were the correct version that had been approved by Configuration Control department.

Therefore, the development methodology was once again modified to incorporate the approval process of multimedia document. The modified methodology was then used to develop Multimedia MPS (MM-MPS). The results from the evaluation of MM-MPS showed that the new development methodology suited all the requirements of the stakeholders of the application, including the end users and the content providers (manufacturing process experts).

The refined methodology is depicted in Figure 74.

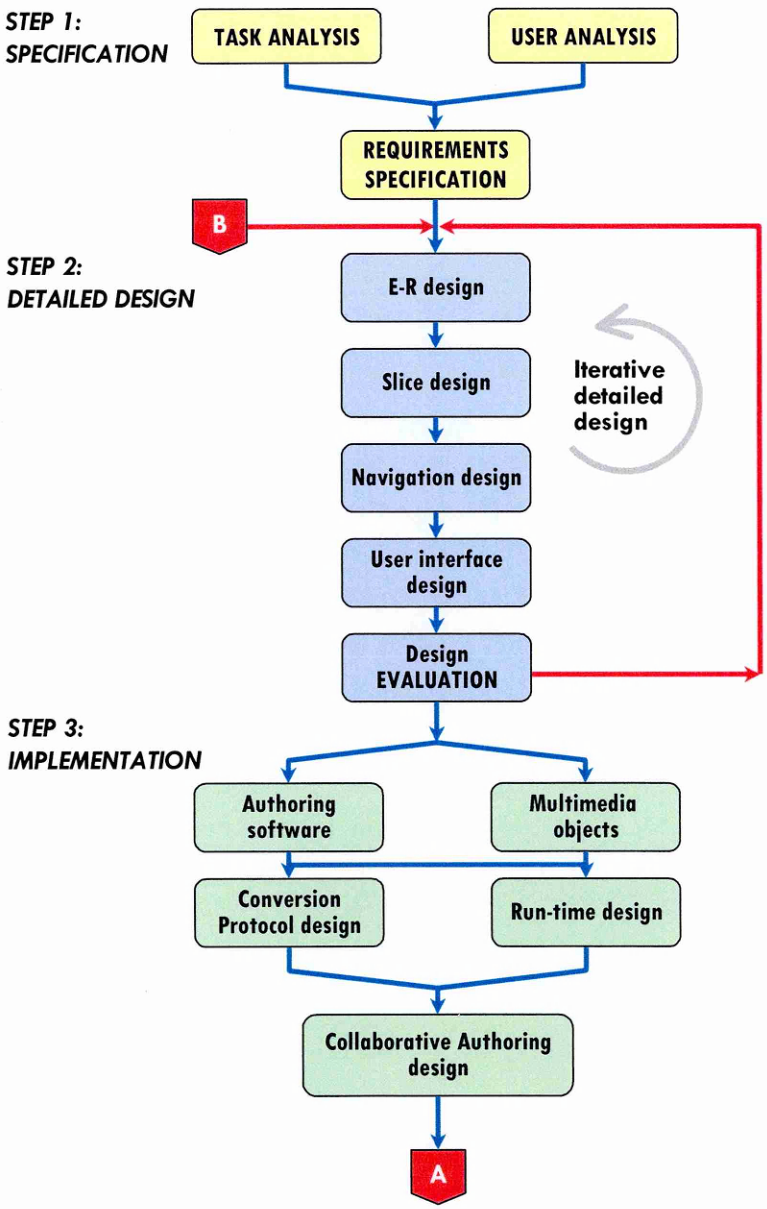


Figure 74 – Refined development methodology

Compared to the proposed methodology, steps 1 and 2 remain the same. However, step 3 has been modified. In the proposed methodology, the output from step 3 (implementation) is the actual hypermedia task support system, whereas in the refined methodology, the output from step 3 is the *collaborative authoring environment*. This include templates and other administration pages that allow the content providers to create the multimedia objects, edit them and publish the content of the task support system including multimedia objects using the upload mechanism provided within collaborative authoring.

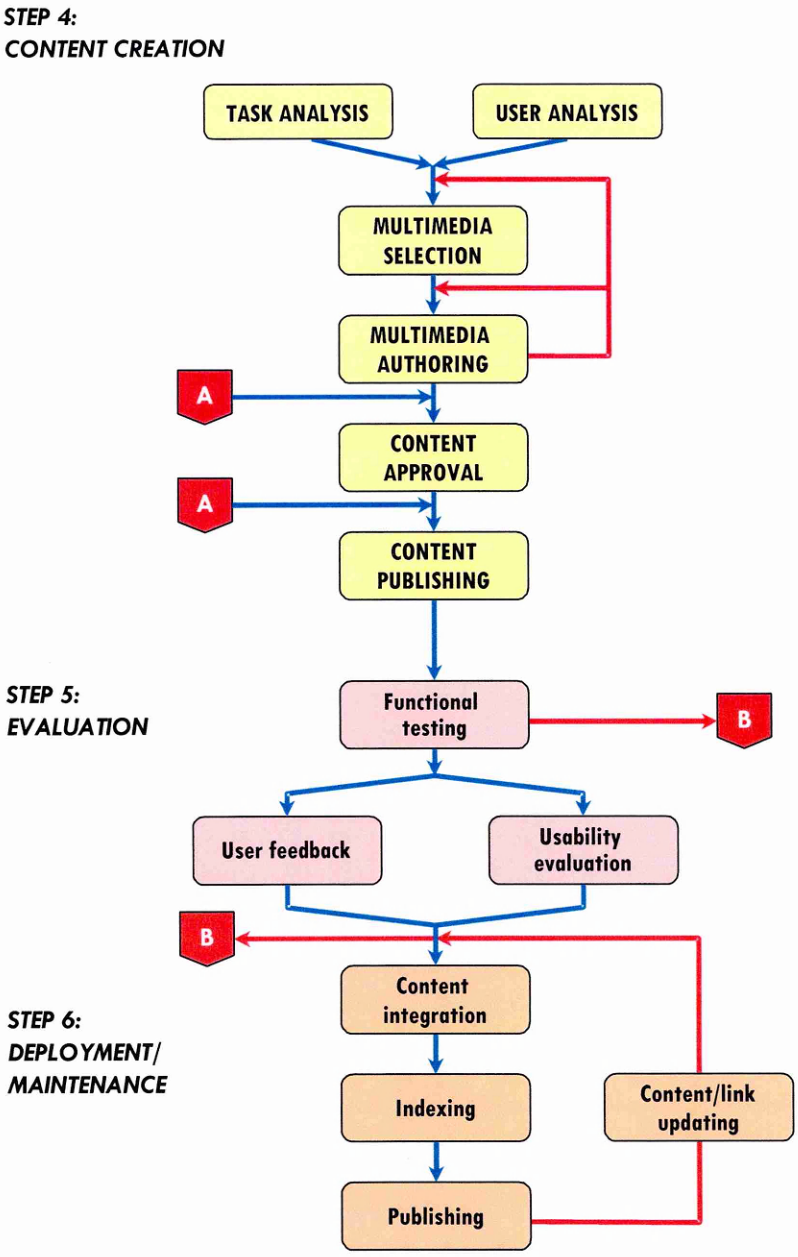


Figure 74 – Refined development methodology [*continued*]

Unlike the proposed methodology, the refined methodology emphasises the role of content providers in developing the content for the task support system. It is their responsibility to ensure that the content is accurate and usable.

6.6 Summary

This chapter has described the case studies carried out in the sponsoring company. The aim of the case studies was to investigate whether the proposed development methodology has fulfilled the success criteria from both the end users' and the manufacturing process experts' points of view.

Four industrial hypermedia task support systems were developed using the proposed methodology. Each task support system was evaluated using the research instrument formulated in Chapter 5. The results from the evaluation determined whether or not the proposed methodology needed to be modified.

Each case has demonstrated the need for modification of the proposed methodology. For example, in Case 2 (ManTIS reference mode and training mode), the lack of currency was due to the absence of the mechanism that allows document authors (content providers) to update the content quickly and easily.

The refined development methodology incorporates the collaborative authoring environment that facilitates the TSS content providers not only to create multimedia objects, but also to update the current content. The refined methodology was also evaluated by a number of content providers (*secondary users*). The results indicated that they accepted the development methodology.

7 Confirmatory Case Studies

This chapter describes the last stage of the research programme. It discusses the confirmatory case studies that were carried out to investigate the extent to which the refined methodology, developed at the end of the case studies, might be relevant and applicable to companies other than the sponsor. Therefore, it was expected that, through these confirmatory studies, the generality and robustness of the methodology could be achieved.

7.1 Introduction

The confirmatory case studies involved a number of companies, chosen because of their appropriateness for the research purposes and their availability. Four companies across multiple sectors participated in the studies, all of them having developed and implemented various information systems using hypermedia/web technology. They were: a high technology vacuum manufacturer; an automotive engine manufacturer; a business unit of a company producing technical manuals for both military and commercial purposes; and a central service of a university which produced hypermedia/web applications.

The confirmatory case studies were all conducted through a one-day company visit. Both quantitative and qualitative approaches were used during the visit, in order to carry out the study in a flexible manner. A questionnaire was sent to the participants prior to the visit, giving an overview of the aims of the confirmatory case studies. During the visit, the participants were also interviewed about various aspects of the hypermedia systems developed. Finally, the methodology and the success model developed during the research were presented to the participants. They were asked to comment on the applicability of each stage of the methodology to the company.

Some of the participants had not completed the questionnaire, and only filled it in just before the interview started. In these cases, the methodology to be validated was not presented to the participants at the beginning of the interview, in order to avoid the risk of bias and to isolate the findings. This ensured an objective comparison between the methodology developed in this research and that used by the companies.

The questionnaire consisted of two sections. The first part asked the participants to give their comments about the *extent to which the elements of the development methodology, embodied in 10 questions, had been adopted during the development of their hypermedia applications*. Examples included confirmation about the usage of tools or techniques they might have used. The last section comprised four questions aimed at obtaining their opinion about the success criteria for the development and implementation of hypermedia application in their companies. The questionnaire which was used during the confirmatory case studies is given in Appendix B.

The interview, which started as soon as the participant handed over the questionnaire, attempted to identify the type of hypermedia application developed by the company

visited and to expand on the responses to the questionnaire. The aim was to obtain further explanation as to the 'how' or 'why' of the participant's answer to each question.

After the initial interview, the development methodology and the success model were presented to the participants, with an explanation as to how to use the step-by-step development methodology, both in tabular and flowchart formats. The availability of supporting templates and forms that could be used during the development of hypermedia application was also demonstrated. Finally, the participants were invited to comment as to whether the methodology was relevant and appropriate, and consequently useful to them. If they felt it was not, they were asked to explanation why.

The following sections discuss the details and the outcomes of the confirmatory case studies. In each case, the discussion is split into two parts. The first part describes the hypermedia application developed by the company, along with the development process adopted. The second part provides analysis of the questionnaire and the interview conducted during the company visit. The analysis highlights the features of the proposed development methodology that were most and least adopted by the company.

A cross-case analysis is provided towards the end of the chapter, to outline both similarities and differences in the extent to which the companies had adopted the features of the hypermedia development methodology.

7.2 Company A: high technology vacuum manufacturer

The company is a leading manufacturer of vacuum pumps for industrial and scientific applications. The manufacturing facilities are spread across a number of countries on four continents. Given the predominantly market-driven situation, the company has always tried to be the first in the industry sector with an e-commerce capability to market its products. This is expected to increase sales revenue, promote more business channels and increase customer relationship.

The e-commerce system developed by the company is capable of building competitive advantage by taking a step forward along with the overwhelming growth on the Internet. The system provides the following features:

- An online product catalogue for more than 9,000 vacuum products and spare parts,
- A tool to configure the product according to its product matrix,
- Online order placement and payment and, more importantly,
- Post-sale documentation support related to the product.

The system was based on an HTML version of the paper catalogue. The structure allowed separation between the presentation and its content, which was stored externally, to enable other applications, such as desktop publisher, to access the content for paper catalogue generation.

7.2.1 The hypermedia application

The company's e-commerce system adopted the typical 3-tier approach, consisting of a dynamically created page using web template, with the data populated from the external data source. The data was stored using a file system structure, and therefore file-naming convention was imposed to ensure data integrity.

The content for the e-commerce application originated from the original paper catalogue. Therefore, the original data files in Adobe FrameMaker format had to be deconstructed to create content for the catalogue data warehouse. The content was organised into 8 categories: Product Overview, Feature & Benefits, Applications, Spares & Accessories, Technology, Technical Data, Performance Data and Product Dimensions. These categories were the main entry points to the catalogue, represented by navigation bars, as shown in Figure 75.

A Master Data Spreadsheet (MDS) was created, along with all the necessary support information, including performance charts. These charts were enhanced with colour to help the viewer define product performance. All files were renamed using a structured file naming convention and were converted into MS Word RTF (Rich Text Format).

Finally, additional data, such as photographs, were re-sampled and re-sized according to the specifications.

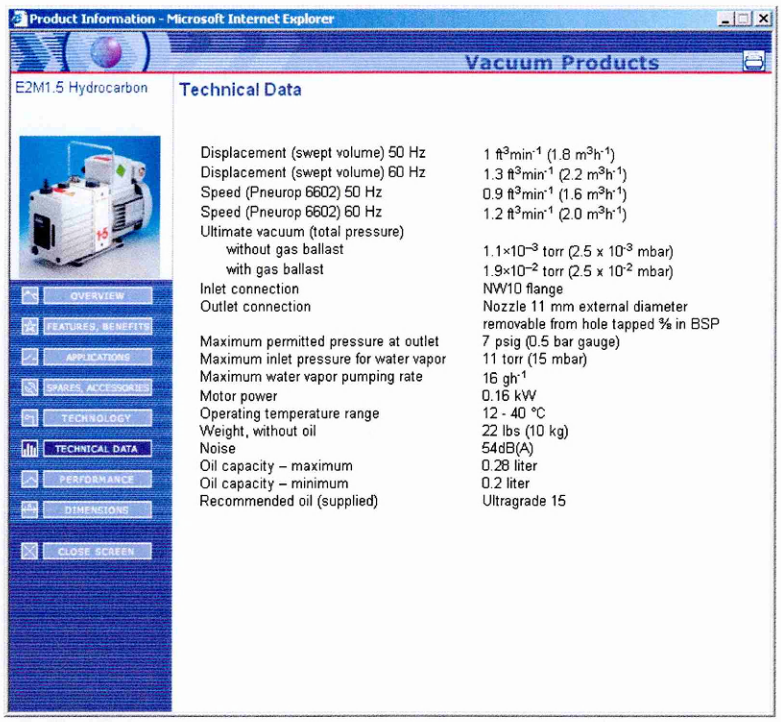


Figure 75 – Product catalogue

In order to update and maintain the content of the e-commerce catalogue, an administration system was developed, with two main features related to *content management* and *content synchronisation*.

Content management comprised a workflow system that allowed the content owners (usually product managers) to download the product description files, edit them, amend the graphical information and upload the files back to the server, using a common platform or user interface. *Content synchronisation* dealt with the capabilities of the system to link a product number to its category, regenerating the required HTML files and finally setting the product data live. Figure 76 and Figure 77 show the flow for editing or updating the content of the catalogue.

To edit the content, the relevant text file in RTF format was opened using MS Word 2000, directly accessible from the administration page. The file was automatically saved in RTF and HTML, when the administrator had completed the changes. The resulting HTML file name and the links from the revised HTML file to SKUs or Categories were imported into MS SQL database. A new file could be published to the server using the upload page, integrated from the administration page. The page provided a file search, so that the administrator could locate the file easily. Once uploaded, the file appeared on the file list and was ready to be activated.

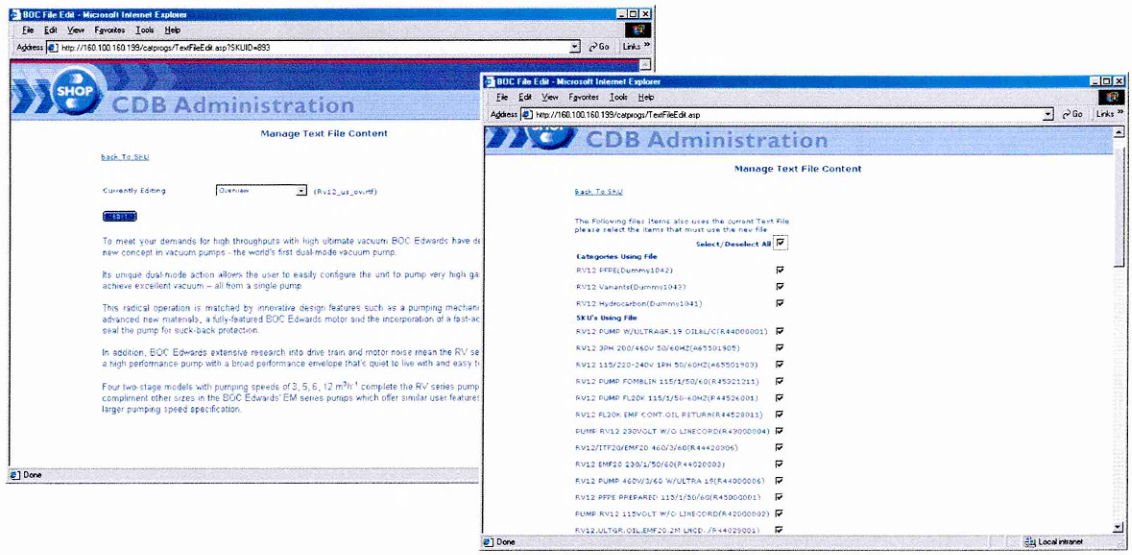


Figure 76 – Text and linking pages

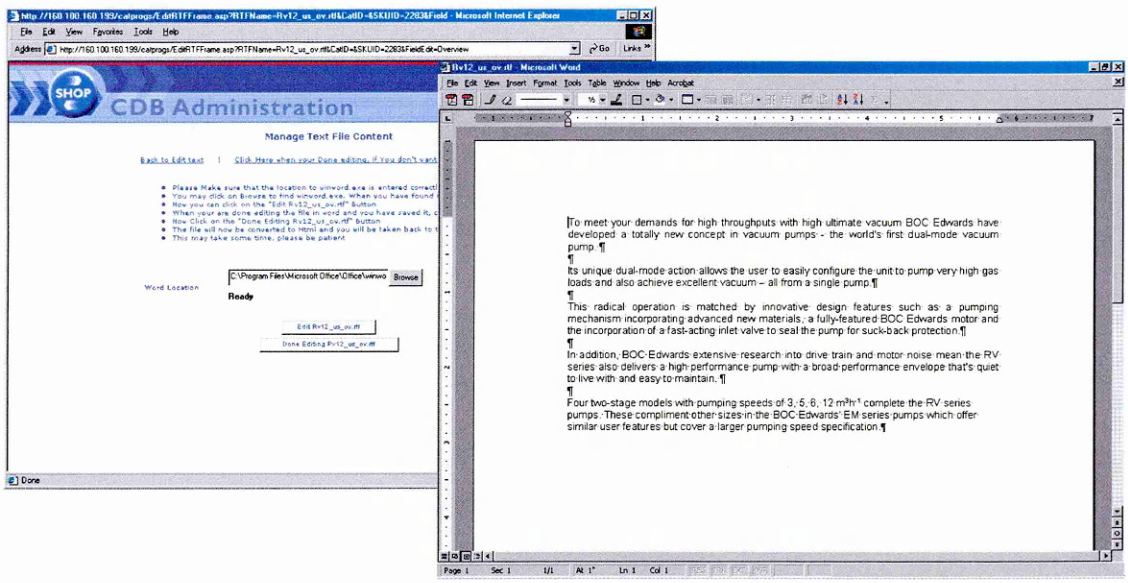


Figure 77 – Edit and management pages

7.2.2 Case analysis

During the study, the Technical Publicity Manager, responsible for the generation and maintenance of technical publications, electronic data and e-commerce catalogues, was interviewed. The purpose of the interview was to expand on the responses from the questionnaire that had previously been filled out.

Table 35 shows the results of the questionnaire for company A.

Table 35 – Features of development methodology adopted by company A

Features within development methodology	Adopted
Formal methods to elicit user requirements and specifications for the application	●●●●●●
Stakeholders' (users, process experts, developers) involvement during design process	●●●●●○
The use of formal design techniques: in-house methodology, SSADM, SDLC, OO, E-R, etc.	●●●●●●
Sole responsibility of the developer to generate, publish and maintain the content	●●●●○○
Separation of content and presentation	●●●●●●
Mechanism to allow other people to contribute the content easily	●●●●●●
Documentation control and content approval of the application	●●○○○○
Methods and strategies to maintain complex hyperlinks and to ensure the content remains up to date	●●●●●○
Adoption of formal evaluation techniques to ensure usability of the application	●●●○○○
Provision of other in-house success measures	●●●●●○

As can be seen in the above table, the documentation control and content approval processes were not well established within the entire development cycle. The accuracy of information was solely ensured through 'informal' checking procedures, for example, using manual data naming convention and Master Data Spreadsheet (MDS).

Similarly, no formal evaluation techniques were adopted by the company to ensure the usability of the e-commerce system. However, according to the company, it was somewhat more appropriate to measure the success of the application from the business point of view, rather than focusing solely on its usability.

Nevertheless, the company claimed that their methods and approaches in developing their e-commerce system were appropriate to ensuring a successful system. This was indicated by the large number of people visiting the site and, consequently, a substantial amount of orders received directly through the e-commerce system.

7.3 Company B: automotive engine factory

The company is an automotive engine factory producing four cylinder petrol engines, from 1.6 to 2.0 litres capacity. The factory has implemented total quality management to ensure the best quality of the products, through conformance in the manufacturing process. Therefore, every manufacturing process has to be well documented in a Quality Management System (QMS).

7.3.1 The hypermedia application

The company developed an electronic QMS known as Multimedia QMS or MM-QMS, using web-based technology (Figure 78). The system has overcome many problems associated with paper-based QMS, such as accessibility, tedious cross-referencing and lack of document currency.

Web technology is used to deliver MM-QMS to the factory, enabling a wide range of users, from shop floor operators to factory managers, to access the QMS from their workplace. The web also allows an open architecture, where document linking of various types is possible. HTML is chosen as the main technology to distribute MM-QMS through the company's Intranet. This results in more flexible navigation, enabling access to relevant information about the manufacturing processes in operation, through a number of different access routes into the MM-QMS.

The content of MM-QMS originated from the existing paper-based QMS, written by the *process owners* (who understood the process well), using a standard word processing package, i.e. Microsoft Word. Although HTML editors, such as Microsoft FrontPage, are commonly use to create HTML files, apparently the learning curve when introducing this package to process owners must be taken into account. The introduction of a new technology often increases the workload and demands a lot of time and effort to learn.

For that reason, the MM-QMS was written in MS Word format, and the process owners published the Word documents to HTML. This was made possible by a built-in MS Word feature called the HTML assistant. The approach was found to be cost effective and efficient, because the process owners did not need to learn how to use the HTML editor.

However, there was no specific style format for the QMS presentation and, therefore, no control of document layout and consistency. To overcome this, a template consisting of a set of styles compatible with the HTML specification was developed by an IT expert. It was mandatory that this template be used every time the process owner wrote a document for MM-QMS. In addition, a macro was created to instruct the HTML assistant to convert the MS Word documents into HTML, with embedded style.

While working with MS Word, the process owners also defined the hyperlinks from the table of contents to the body of the QMS. These links were referred to as the *internal hyperlinks*. The appearance (font name, font size, colour, alignment, etc) of the style was controlled by the Cascading Style Sheet (CSS). The CSS was a text file that consisted of a set of rules on how a style was defined. Linking the CSS file to HTML files on the server ensured that the HTML files were consistent in look and feel. Another advantage was that changes in the CSS file would affect all the HTML files in the server, and this reduced a lot of maintenance effort.

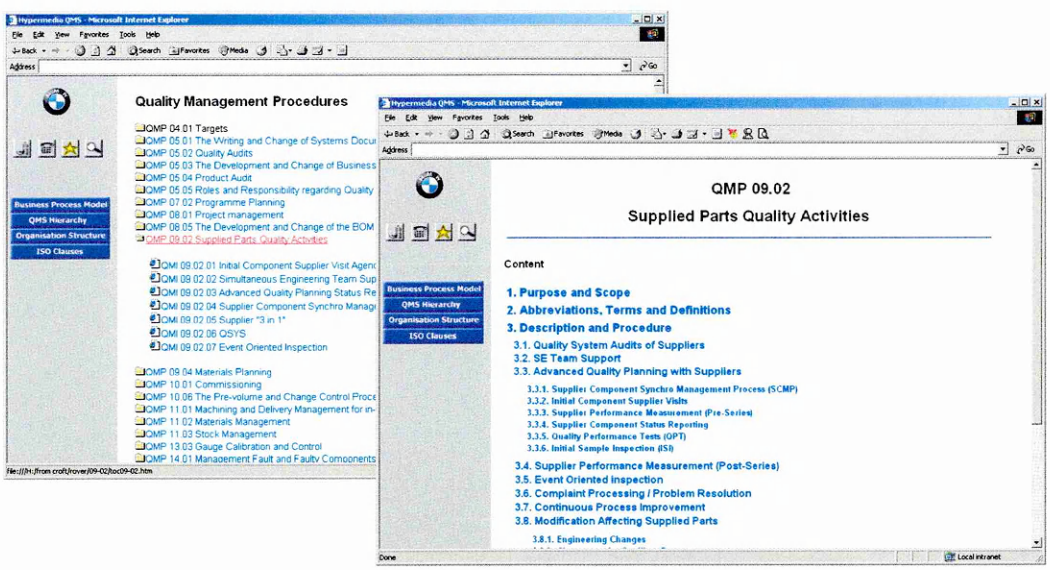


Figure 78 – Multimedia Quality Management System

7.3.2 Case analysis

During the case study, the Quality Manager was interviewed during a one day visit to the company in Hams Hall, near Birmingham. The Quality Manager was responsible for ensuring that the content of MM-QMS was up to date and accessible through the company's Intranet.

Unlike company A, the questionnaire was filled in during the visit but prior to the interview session. The results are summarised in Table 36.

Table 36 – Features of development methodology adopted by company B

Features within development methodology	Adopted
Formal methods to elicit user requirements and specifications for the application	●●●●●●
Stakeholders' (users, process experts, developers) involvement during design process	●●●●●●
The use of formal design techniques: in-house methodology, SSADM, SDLC, OO, E-R, etc.	●●○○○○
Sole responsibility of the developer to generate, publish and maintain the content	●●●●●●
Separation of content and presentation	●●●●○○
Mechanism to allow other people to contribute the content easily	●●○○○○
Documentation control and content approval of the application	●●●●●○
Methods and strategies to maintain complex hyperlinks and to ensure the content remains up to date	●●●●○○
Adoption of formal evaluation techniques to ensure usability of the application	●●○○○○
Provision of other in-house success measures	●●●●●●

The results indicated that no formal design techniques were adopted during the development of MM-QMS although, to a certain extent, some of the features existed within the application, for example the use of document templates. This was perhaps due to the Quality Manager being unaware that such design methods exist within the hypermedia and software engineering fields. As a result, the MM-QMS did not allow the content provider to create and update content.

As mentioned earlier, the content providers of MM-QMS, i.e. the process owners, wrote their documents using MS Word, created the HTML version of that document, and passed the HTML files to the hypermedia developer to be linked and published on the server. It was also confirmed from the questionnaire that the developer was solely responsible for publishing the content.

The company did not incorporate formal evaluation techniques to ensure the usability of the application. Apparently, according to the company, the purpose of the MM-QMS was not solely for the provision of QMS online on the computer screen, but also to

support continual improvement activities. Therefore, usability, according to the company, was not the primary success measure of the MM-QMS.

Nevertheless, the company adopted formal methods, such as quality function deployment, to elicit user requirements and, more importantly, stakeholders of the application were consulted during the development of the application. This ensured that the customer's voice was considered and incorporated into the application. As the MM-QMS was a quality documentation system, it was imperative that approval and control of the document was put in place and, indeed, the company confirmed that it was.

7.4 Company C: technical manual producer

The company is renowned for providing *whole life support* for military equipment, for command and control applications. In addition, the company also produces commercial technical documentation, applying the experience gained from continual provision of similar services to the Ministry of Defence (MoD) and the aerospace industries. The advancement of multimedia technology, combined with the ever-increasing number of in-house authors from different engineering backgrounds, has led the company to expand its competencies towards electronic publishing services and multimedia capability, to enhance its services to customers.

In this case study, the company was chosen for its involvement in the development of an interactive task support system for Oxford Magnet Technology (OMT), a leading manufacturer of cryogenic magnets. What is interesting in this case study is the way that company C, as a consultant to OMT, introduced a range of electronic technical manuals to help OMT's in-house engineers and field service engineers carry out maintenance tasks. The problems faced by OMT prior to the introduction of the electronic technical manuals included the burden on the service engineers of a large amount of paper-based documents that contained too much irrelevant information. To complicate matters, there was a high frequency of contact from the field service engineers to the service helpdesk, despite the availability of the paper manuals.

7.4.1 The hypermedia application

The high overhead costs for manual production, combined with more complicated procedures for field service engineers, meant that OMT needed to provide field service engineers with an effective and efficient way of accessing the information relevant to their task. One of the most prevalent strategies was the adoption of multimedia technology to develop Interactive Service Manuals (ISM) (Figure 79), delivered through various means of low maintenance resources, such as laptops.

In response to OMT's standard, quality and cost requirements, the ISM was developed to incorporate the following features: hyperlinks, zoom/pan images, animated sequences, vector graphics for displaying wiring diagrams and a search engine. Animations and zoom/pan features were built using Macromedia Flash, whereas Java applets were used to create a search engine to allow quick and easy information retrieval. In order to ensure its accessibility and availability, ISM was delivered through CD-ROMs and the Intranet.

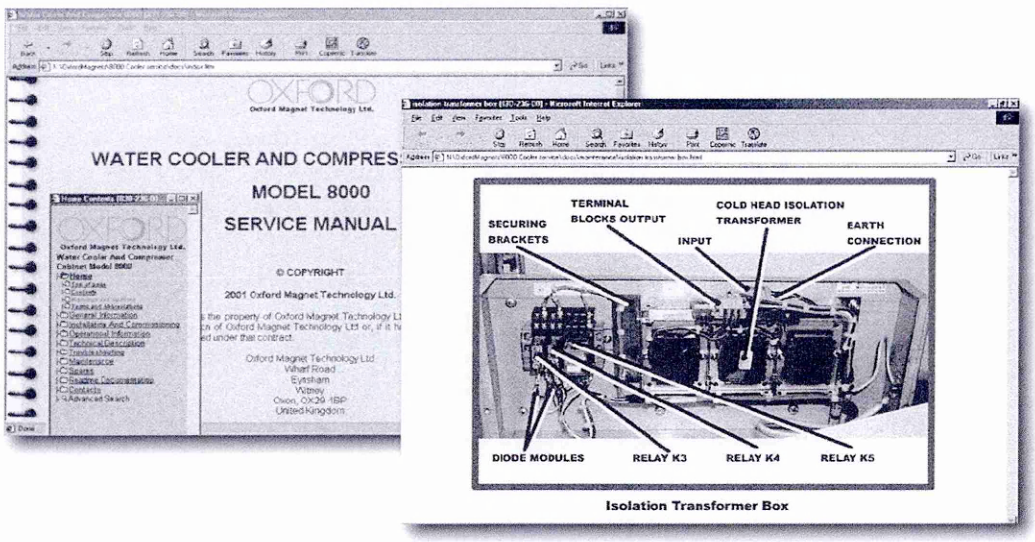


Figure 79 – OMT's Interactive Service Manual

7.4.2 Case analysis

The case study was carried out through a one day company visit to the site in Malvern, Worcestershire. During the visit, the Business Development Manager of company C, who was also the head of the business unit responsible for producing commercial multimedia applications, was interviewed. As with company B, the questionnaire was also filled out on site prior to the interview. The results are shown in Table 37.

This table shows that the content and presentation of OMT's Interactive Service Manual were not separated, meaning that hyperlinks, as well as document formatting such as styles, were embedded within the content. The implication was that only developers who understood the technical aspects of the application would be able to create and update the applications. It was, therefore, not surprising that generating and updating the content of the ISM was the sole responsibility of the developer. Consequently, it was very difficult for the content authors from OMT, who might not be familiar with the hypermedia technology, to update the content.

Table 37 – Features of development methodology adopted by company C

Features within development methodology	*Adopted*
Formal methods to elicit user requirements and specifications for the application	●●●●○○
Stakeholders' (users, process experts, developers) involvement during design process	●●●●●○
The use of formal design techniques: in-house methodology, SSADM, SDLC, OO, E-R, etc.	●●●●●○
Sole responsibility of the developer to generate, publish and maintain the content	●●●●●●
Separation of content and presentation	●●○○○○
Mechanism to allow other people to contribute the content easily	●●○○○○
Documentation control and content approval of the application	●●●●●●
Methods and strategies to maintain complex hyperlinks and to ensure the content remains up to date	●●●●○○
Adoption of formal evaluation techniques to ensure usability of the application	●●●○○○
Provision of other in-house success measures	●●●●●○

Nevertheless, company C confirmed that a formal change control of document and approval mechanism was incorporated throughout the development of ISM. This included the use of a sign-off sheet for text and images used in the application. During the approval process, company C established a communication link to OMT, to check progress and to ensure a tighter control of the quality assurance process by OMT.

7.5 Company D: a central service

The company is one of the central services within a university, producing web-based systems used by other departments. One of the products developed by the company is the job advertisement site for the Personnel department (called 'Application A' hereafter). Previously, the Personnel department wrote the job advertisement contents, and the company converted it manually into a number of static HTML pages. As the length of time taken to make appointments was critical, it became apparent that the process was not adequate. In addition, the documents produced by the Personnel department did not use the guidelines prescribed by the university, resulting in inconsistency of the information presented to potential applicants.

In another project, the company also developed a web-based system for short-courses offered by the university (called 'Application B' hereafter). This was an on-line system where the applicants could browse the schedule of courses and enrol electronically. This system was also time-critical, because the courses had fixed start dates throughout the year. As the courses were offered by a number of departments within the university, they had to supply details of the course contents, the syllabus and the cost involved, to the company. In addition, any updates and revision had to be done by the company. This slowed down both publishing and updating processes.

In order to solve these problems, the company reengineered the old publishing and authoring process from a static, manual approach to a more dynamic approach. It was realised that the flow of information from authoring to publishing was not smooth. Therefore, the company initiated a new approach by employing a web-based content management system (CMS). The company used a package called Spectra to manage the documents.

7.5.1 The hypermedia application

In both Applications A and B, the CMS required the content provider to create the content using a predefined template. This allowed the content to be separated from the layout, and it also released the developer to format the document in such a way that it followed the corporate standard of the university. Neither the personnel officers nor web developers created 'free form' HTML pages. Instead, using the user interface provided, content providers entered the information content directly onto the system. To minimise error and to make it foolproof, the user interface did not allow the content providers to enter free text in certain types of field. Instead, they had to select from a drop down menu. The system formatted the content and displayed it accurately and consistently.

From the above, it is clear that the company employed the model of a dynamic authoring system, where the developers did not directly deal with the content of the document. Instead, they enabled the content providers to post the content to the CMS, by providing an intuitive, easy to use user interface.

7.5.2 Case analysis

During the case study, the company's *web developer* was interviewed. The interview was guided by the completed questionnaire, the results of which are shown in Table 38.

The table indicates that the company did not use documentation control and formal approval process techniques in their applications, simply because their content was not critical and did not imply a significant risk should it be slightly inaccurate.

Furthermore, the company did not considered applying formal usability evaluation to their applications. Compared to task support systems, the applications were clearly much simpler. However, the company did use sound techniques, for example, pre-emption of potential errors, to indirectly ensure the usability of the application.

Table 38 – Features of development methodology adopted by company D

Features within development methodology	*Adopted*
Formal methods to elicit user requirements and specifications for the application	●●●○○○
Stakeholders' (users, process experts, developers) involvement during design process	●●●●●●
The use of formal design techniques: in-house methodology, SSADM, SDLC, OO, E-R, etc.	●●●●●●
Sole responsibility of the developer to generate, publish and maintain the content	●●●○○○
Separation of content and presentation	●●●●●●
Mechanism to allow other people to contribute the content easily	●●●●●●
Documentation control and content approval of the application	●●●○○○
Methods and strategies to maintain complex hyperlinks and to ensure the content remains up to date	●●●●●●
Adoption of formal evaluation techniques to ensure usability of the application	●●○○○○
Provision of other in-house success measures	●●●●●○

7.6 Cross-case analysis

The previous sections have outlined the analysis of each case study, as indicated in Table 35 to Table 38. The key features of the development methodology identified in each case were converted into numbers and are summarised in Table 39.

Table 39 – Comparison of the features from the case studies

Features within development methodology	Company				
	A	B	C	D	TOTAL
Formal methods to elicit user requirements and specifications for the application	6	6	4	3	19
Stakeholders' (users, process experts, developers) involvement during design process	5	6	5	6	22
The use of formal design techniques: in-house methodology, SSADM, SDLC, OO, E-R, etc.	6	2	5	6	19
Sole responsibility of the developer to generate, publish and maintain the content	4	6	6	3	19
Separation of content and presentation	6	4	2	6	18
Mechanism to allow other people to contribute the content easily	6	2	2	6	16
Documentation control and content approval of the application	2	5	6	3	16
Methods and strategies to maintain complex hyperlinks and to ensure the content remains up to date	5	4	4	6	19
Adoption of formal evaluation techniques to ensure usability of the application	3	2	3	2	10
Provision of other in-house success measures	5	6	5	5	21

7.6.1 Requirements specification

With the exception of company D, all the companies that participated in the study used at least one formal method to elicit user requirements and specifications for their hypermedia applications. Companies A and C used qualitative approaches, such as interview and brainstorming, whereas company B used a technique called a requirement matrix, which is a simplified version of Quality Function Deployment [Akao, 1990].

7.6.2 User involvement during design process

All companies involved the stakeholders of the hypermedia application – including the end users – during the design process. They all recognised the importance of reducing the risk of failure by continually gathering feedback from the end users at the early stages of the development phase, thereby reducing the need for major modifications to the application at later stages.

7.6.3 Formal design methods

Companies A, C and D adopted formal design techniques to develop the hypermedia application. Company A used a supply chain management model integrated to the 3-tier e-commerce system. The e-commerce system itself used a relational database model to separate presentation and business logic. Company C implemented the semi-structured System Development Life Cycle (SDLC) [Powell *et al*, 1998] and project management. The method focused on a close interaction with the customer. Company D used a web development tool adopting object-oriented techniques [Booch, 1994]. Company B, to a certain extent, followed a formal method, for example by separating the document formatting and the content using Cascading Style Sheet (CSS). On the other hand, all the companies used the informal method and design heuristics to carefully design the layout and appearance of the application.

7.6.4 Reliance on hypermedia developer

Both companies B and C relied on the hypermedia developers to update the content and links within the hypermedia applications. It is interesting to note that company C produced multimedia technical documentation for its customers. It was implicitly revealed that the nature of the business in company C might have influenced the design strategy, allowing the content of the application to be modified only by company C. In company A, although content generation could be done by people other than the hypermedia developer, the publication of the product catalogue remained under the control of the Technical Publicity Manager, who was also the system administrator.

7.6.5 Content management system

Both companies A and D adopted a full content management system, separating content and presentation. The content of the application was stored externally and was assembled *on-the-fly,* only when required. Company C, on the other hand, did not separate content and the presentation, which resulted in a more time consuming updating process, because it usually required a hypermedia expert to carry out the job. Nevertheless, perhaps this is the business model typical consultancy companies must follow to engage with their customers more tightly. In company B, the response to this particular question was somewhat close to ambivalent.

7.6.6 Other people to contribute content

The advantage of a content management system is the possibility that other people (content providers) can contribute the content of the hypermedia application. Administration pages created by companies A and D enabled the content providers to edit the content online, without having to be expert in hypermedia/web technology. Not surprisingly, the applications developed by companies B and C, to a large extent, did not allow people other than the hypermedia experts to modify the content.

7.6.7 Document control and content approval

With regards to documentation control and content approval, companies B and C confirmed the use of formal methods. Company B used web technology to deliver quality documents. It was therefore not surprising that company B implemented a document control and approval system for any procedure released in the factory. Similarly, company C also adopted a formal approval method to ensure that the content of the Interactive Service Manual (ISM) delivered to OMT had been checked and was error free. On the other hand, company A only applied an informal process to control the content of the catalogue, which was the responsibility of the Technical Publicity Manager.

7.6.8 Strategy for links maintenance

All companies seemed to have a strategy to maintain the links within the application. Companies A and D, for example, utilised the features in a content management system to maintain the links, whereas companies B and C used an ordinary links management system with built-in web development tools. In all cases, the strategies included consistent use of file name and the hierarchy of the documents.

7.6.9 Formal usability evaluation

Perhaps this is the most interesting result. None of the companies used a formal evaluation method to ensure the usability of the hypermedia application. The reasons varied from case to case.

Company A asserted that usability was of high importance in any IT application. However, for the e-commerce system, it was more appropriate to consider other factors, such as the performance of the system, the content of the catalogue and the maintainability of the data, all of which had been proven to give a better business result. In company B, the MM-QMS was intended to address problems associated with accessibility, cross referencing and the currency of the quality documents which, to a certain extent, meant the loss of some aspects of end users' usability. According to company B, success was believed to be dependent on the extent to which the QMS could support continual improvement activities. Company C focused more on fulfilling

the standard on product specification and reducing cost of ownership, by allowing the author and the hypermedia expert to work together on site, ensuring the accuracy of the content. Finally, company D believed that adequate training for the users of their applications, as well as the robustness of the applications that could catch errors and thus ensure data integrity, were as important as the usability issues.

7.6.10 Success measures

All the companies had set a measure for the success of their hypermedia applications, throughout their entire life cycle, from design to implementation and to maintenance.

Company D defined the success measures during the design, implementation and maintenance of the web system. Clear instruction on the features of the application, knowledge of the end users and the hardware/software, and careful selection of development tools were among the success measures during the design stage. Thorough training, management support and authority to make decision were crucial during the implementation stage. Finally, anticipation of possible errors, through a foolproof mechanism, was required for maintenance.

For company C, close and continual interaction between technical author and developer was considered to be the success factor during both the design and implementation stages. The success of the maintenance stage was achieved through the reduction of cost of ownership, in which amendment and revision could be done through a solid cooperation between company C and its customers, including OMT.

Company B did not split the success measures from the design, implementation and maintenance aspects. Rather, the company emphasised that the ultimate goal of the MM-QMS would be achieved if it could contribute to improved efficiency of the processes and speed up problem resolution. To achieve that, MM-QMS had to be '*fresh at all times*' or, in other words, the content had to be up to date, and hence useful for the users. However, according to the company, the above could lead to a high maintenance cost if the system was not well designed. Therefore, a structured approach had to be taken into account during the development of the application.

Company A strongly believed that a clear and focused methodology, that took into account both long- and short-term business goals, would ensure the success of the hypermedia application (in this case, the e-commerce system). The implementation and deployment strategies for the e-commerce system adopted a proven technology which was combined with knowledge of the products. Maintenance was of high importance and was a key element within the e-commerce system. Therefore, a system was specifically developed to manage the rich content and to reduce maintenance effort.

7.7 Confirmatory analysis of the development methodology

The results of the confirmatory analysis of the development methodologies used by the companies visited are listed in Table 40.

From the table, it can be seen that the first step of the development methodology, i.e. requirements specification, was fully supported by all companies, with the exception of company D. This result is not unexpected, because it is in fact the step that also exists within the development methodologies proposed by Powell *et al* [1998], Lowe and Hall [1999], Crowder *et al* [2000] and Olsina [1998].

In step 2 (detailed design), only companies A and D fully supported the use of both *E-R design* and *slice design*. Company B did not see the need for using these techniques within their hypermedia application, whereas company C partly supported this. The latter asserted that E-R design would only be used when developing a rich content, database driven application, which would be more difficult to create without such an approach. Not surprisingly, all the companies supported both *navigation design* and *user interface design*, although to a certain extent they were not aware of these notations, as they are used in the hypermedia literature. All of them used standard storyboarding techniques to document navigation and hyperlinks. They also incorporated design heuristics to create the user interface, particularly when designing for consistent look and feel. Once again, only company D fully confirmed the use of evaluation during the design process.

Only companies A and D fully confirmed all the stages within step 3 (implementation). This was because the hypermedia applications developed by both these companies used dynamic, *on-the-fly* document generation, where content was populated into a template from an externally stored database. Companies B and C clearly did not use a method to separate content and presentation, and therefore they were concerned only with the actual authoring (and programming) of the hypermedia applications.

Interestingly, none of the companies fully supported usability testing and evaluation as proposed in the development methodology; only functional testing was considered important.

Content approval and sign-off processes were highly important for both companies B and C, due to the fact that both applications contain quality documents, which need to be properly reviewed and approved before being published.

Finally, all companies supported the deployment and maintenance step, as proposed in this research, with an additional option for integrating a search mechanism to the system. It was apparent that all companies had incorporated a search engine in their applications.

Table 40 – Confirmatory analysis of development methodology

Step	Description	A	B	C	D
1	***Requirements specification***				
	▪ Formal method: task analysis, QFD	●	●	●	◐
	▪ User involvement: user analysis	●	●	●	●
2	***Detailed design***				
	▪ Entity-Relationship design	●	○	◐	●
	▪ Slice design	●	○	○	●
	▪ Navigational design	●	●	●	●
	▪ User interface design (design heuristics)	●	●	●	●
	▪ Design evaluation	○	◐	◐	●
3	***Implementation***				
	▪ Authoring tools selection	●	●	●	●
	▪ Conversion protocol design	●	○	○	●
	▪ Run-time design	●	●	●	●
	▪ Template generation	●	○	○	●
4	***Content population/creation***				
	▪ Selection of supporting multimedia objects	●	◐	◐	●
	▪ Content authoring	●	◐	◐	●
	▪ Content approval (sign off & document control)	◐	●	●	○
5	***Evaluation***				
	▪ Functional & usability testing	◐	◐	◐	◐
6	***Deployment and maintenance***				
	▪ Content integration	●	●	●	●
	▪ Publishing	●	●	●	●
	▪ Content/link update	●	●	●	●

● fully confirmed
◐ partly confirmed
○ not confirmed

7.8 Summary

This chapter sets out the case studies undertaken to confirm the development methodology for a hypermedia task support system. The studies were carried out through company visits, during which the participants were given questionnaires and were asked to comment on the steps of the development methodology. In addition, semi-structured interviews were conducted to expand on the answers to the questionnaires.

To a large extent, the development methodology was fully confirmed by all companies, with the exception of some stages within detailed design, usability evaluation and content approval. The reasons for this vary from case to case and are dependent on the characteristics and the features of the hypermedia applications.

In hypermedia applications where the content did not contain critical information, i.e. related to health and safety or legal implications, the content approval and sign off process might be skipped. Hypermedia applications where the content needed to be supplied by many people would require a two-step process, in which the template was generated first, followed by content population.

From the confirmatory case studies conducted, it is apparent that all the companies acknowledged that a structured approach – rather than an *ad hoc* one – was required to ensure that the application was also well structured, easy to understand and user friendly. This should lead to the success of the development, implementation and maintenance of hypermedia application.

8 Discussion and conclusions

This final chapter presents the research findings, evaluates whether the findings have fulfilled the research aim and objectives and describes the contribution to knowledge the research has made. In addition, the limitations of the research are presented along with the direction for future work.

The research aim was to formulate a development methodology that can be used to guide the industrialists who intend to develop industrial hypermedia task support systems from conception of idea to systems deployment.

To achieve the above aim, the following research objectives were identified.

1. To develop examples of industrial hypermedia task support systems that satisfied the expectations of all relevant stakeholders of the systems, based on the proposed methodology

2. To investigate the extent to which the proposed development methodology has supported the development of industrial hypermedia task support systems

3. To evaluate the applicability of the development methodology to manufacturing industry in general

4. To provide recommendations to manufacturing organisations that intend to implement industrial task support systems based on hypermedia technology

The availability of an industrial setting has allowed a detailed observation and enquiry to be carried out through a case study-based research methodology. In order to ensure the achievement of the research objectives, a four-stage research programme was adopted.

8.1 Research findings

The following summarises the findings from both the main case studies and the confirmatory case studies.

1. In the sponsoring company, hypermedia developers alone appear to be incapable of providing all the detailed and accurate information required to support manufacturing tasks in a timely manner, due to the volume, variety, and complexity of the tasks that need support. Because manufacturing tasks are defined and re-defined by manufacturing process experts, usually industrial engineers, these people need to be involved in the entire development cycle. This is also confirmed by all companies participating in the confirmatory case studies.

2. This research found that the end user usability alone was insufficient to claim success of an industrial hypermedia system, and indicated that success was also determined by the *development process* of the hypermedia task support systems.

3. This research also found that the success of industrial hypermedia task support systems needs to be judged with respect to two types of usability, which can be termed *primary* usability and *secondary* usability. The ease with which the end users can access content that supports their manufacturing task can be termed primary usability, and this is influenced by both the hypermedia developers and the manufacturing process experts. The ease with which the manufacturing process experts can contribute the content of industrial hypermedia task support systems can be termed secondary usability, and this is influenced by the hypermedia developers.

4. The above is relevant to the principles of quality that imply the ability of the supplier to meet the customer requirements [Oakland, 1993]. The success model proposed in the research supports this. Clearly, the end users are the customers of the manufacturing process experts, while the manufacturing process experts are the customers of the hypermedia developers. Each chain that links the suppliers and the customers is actually a process that transforms a set of inputs into a set of outputs that satisfies the customer expectations. If the chain is broken, the system is considered to have a poor quality, and is therefore unsuccessful.

5. The research has addressed some of the problems faced by academics and industrialists in this field, by providing a step-by-step methodology to create, deploy and maintain industrial task support systems based on hypermedia technology. The methodology proposed in this research has answered the question of how to ensure the success of systems by enabling the creation of structured hypermedia applications that satisfy the requirements from the stakeholders of the system.

6. The development methodology can be used to guide the hypermedia developers (particularly manufacturing/industrial engineers) to create industrial hypermedia task support systems, such as electronic technical manuals, online help systems, and troubleshooting guides. The benefits of adopting the methodology include reduced risk of failure, reduced cost of training and improved control of documentation systems.

8.2 Research findings compared with research objectives

The following sections discuss the comparison between the research findings and the research objectives.

8.2.1 Objective 1: The development of hypermedia TSS

During evaluation of the hypermedia TSS, both the end users and the manufacturing process experts were involved. However, the emphases were different from case to case. Early cases (i.e. Case 1 and part of Case 2) focused more on the success criteria from the end users' point of view rather than from that of the manufacturing process experts. As experience built up, the evaluation focused more on the manufacturing process experts.

In Case 1, the ManTIS simulation mode was well accepted by the end users who were mainly inexperienced operators. They perceived the application to be *easy to use* and *useful*. The correlations between *perceived ease of use* (EOU) and *intention to use* (IU); and between *perceived usefulness* (U) and *intention to use* (IU); to a large extent support the findings from Fakun [2000] and other researchers adopting the Technology Acceptance Model (TAM) [Davis, 1989]. Further analysis indicated that there were correlations between the *task-technology fit* (TTF) and *perceived usefulness* (U); and *perceived easy to use* (EOU). The results also confirmed findings from Dishaw and Strong [1998] that the task-technology fit has a significant correlation to the application usage through *perceived usefulness* (U). The findings imply that the design of task support systems should take into account the task-technology fit dimensions in addition to TAM constructs.

Case 2 demonstrated that –although ManTIS reference mode and training mode showed advantages over paper-based documents in terms of user friendliness, accessibility from the workplace and reliability– they were not extensively used by the shop floor workers, particularly operators and technicians. The long delays in preparing the materials for publication and the reliance upon the hypermedia developer to transfer the materials for ManTIS reference mode and training mode were found to impair the currency of the content. This supports the findings of Kasvi *et al* [2000] whose industrial task support system (called *Task Supporter*) failed because the people from industry could not easily tailor the content to reflect the daily changes in production schedule.

In Case 3, the development methodology was modified significantly to allow the content to be updated easily in order to improve its currency, through the use of collaborative authoring. The task support system developed in Case 3 was evaluated by the end users, whereas the collaborative authoring environment was evaluated by the content providers. The end users' evaluation showed that the task support system had been well accepted by the end users, indicated by the *perceived usefulness* (U) and *perceived ease of use* (EOU) indices. Their comments showed that their understanding

of the tasks had improved through the use of the task support system. The effort required to train new operators was also reduced, allowing supervisors and technicians to spend more time on other development projects. The collaborative authoring environment was also well accepted by the content providers. With the exception of *analysability*, all other *maintainability* dimensions (*reusability*, *changeability*, *stability* and *testability*) exhibited high mean values, resulting in a high *maintainability* index. The low mean value of analysability was due to the fact that the collaborative environment did not provide sufficient control over the multimedia files presented.

Case 4 utilised the development methodology that had been modified and refined as a result from the lessons learnt in previous cases. The task support developed and evaluated in this case, to a large extent, followed the structure of the task support developed in Case 3. The difference, however, was the significant enhancement to the collaborative authoring environment. In this case, the end users were only involved during the preliminary design evaluation. The evaluation of the task support system was focused on the enhanced collaborative authoring environment, involving the content providers as the subjects of evaluation. Once again, the maintainability constructs were employed in the evaluation. The results showed that the collaborative authoring environment had fulfilled the need for the content providers to author the content of the task support system. Not only had it offered easy access to the authoring functions, it had also provided the electronic document approval and sign-off processes.

The adoption of both formal design techniques and design heuristics within the proposed development methodology ensured that the hypermedia TSS developed during the case studies meet the usability criteria, including ease of use, usefulness and maintainability, and satisfy the expectations of all stakeholders of the systems.

8.2.2 Objective 2: Evaluation of the development methodology

The literature review on hypermedia development methodologies has revealed that – amongst the existing methodologies– no single development methodology was appropriate to be adopted 'as-is'; they all required further adjustments to suit the development of industrial hypermedia task support systems. Therefore, a new methodology that is based on the System Development Life Cycle (SDLC) [Powell *et al*, 1998] and spiral model [Boehm, 1988] has been proposed, with many features being more detailed and enhanced.

The first version of the development methodology comprised *five* steps: (1) specification, (2) detailed design, (3) implementation, (4) evaluation and (5) deployment and maintenance. The detailed design, to a large extent, followed RMM [Isakowitz *et al*, 1995] with substantial enhancement, particularly in the design of user interface. The latter incorporated design heuristics suggested by Thüring *et al* [1995], Hardman [1995], Nielsen [1995] and Gygi [1991]. The methodology provides a step-by-step

guide that details the inputs, process, outputs, tools or techniques needed, as well as the individual(s) who will carry out each particular step.

In order to investigate the extent to which the proposed development methodology has supported the development of hypermedia task support systems, evaluations of the development methodology were conducted through a series of case studies. In each case, a hypermedia TSS was developed using the proposed methodology and was evaluated against the proposed success model. The findings from each case became the basis for the refinement of the proposed methodology.

The first refinement was made after the Case 2, in which the feature of the proposed methodology was significantly enhanced with the provision of a *collaborative authoring environment*. This was due to the problems associated with currency of the hypermedia TSS. Collaborative authoring allowed the content providers (usually manufacturing process experts) to contribute the content with minimum assistance from the hypermedia developer. This has improved the speed of content update, which increases the chance that the content of the hypermedia TSS remains current.

The second refinement was made at the end of Case 3, in which content approval was added into the collaborative authoring environment, and consequently was included into the development methodology. *Content approval* or sign-off is required to provide sufficient control over the multimedia files to ascertain that the multimedia objects presented are the correct version and have been approved by the Configuration Control.

The final, refined methodology consists of six steps: (1) *specification*, (2) *detailed design*, (3) *implementation*, (4) *content creation*, (5) *evaluation* and (6) *deployment and maintenance*. The unique feature in the refined methodology compared to existing methodologies is the *collaborative authoring* environment. The 'environment' itself comprises everything necessary to create and update multimedia documents. Examples include image editing templates that work in an image editor package to specify image size, resolution and colour depth. With one click of a mouse, a user can run a macro that edits the picture quickly, and saves it in the appropriate image file format. Without this template, a novice user would need to spend a long time just to create the picture.

In addition, the refined methodology also divides the responsibility and ownership of the development process of the hypermedia TSS between the hypermedia developers and the content providers. Using the refined methodology, the hypermedia developers are no longer required to produce the content. Instead, they facilitate the creation of TSS content by manufacturing process experts through the use of templates, administration pages, etc.

8.2.3 Objective 3: Applicability of the development methodology

In order to investigate the applicability of the development methodology to manufacturing industry in general, the proposed development methodology was validated through confirmatory case studies. Ideally, the confirmatory studies would need to involve as many companies and as many sectors as possible. However, during the confirmatory studies, it was difficult to identify a large number of companies that have fully implemented industrial hypermedia task support systems, and to be considered as best practices. Due to the specific application of the systems developed using the proposed methodology, only four companies were willing to participate.

The results of the studies, to a large extent, indicated that the proposed development methodology was confirmed by all companies, with exception within detailed design, usability evaluation and content approval. The main reasons for these exceptions include the different types of techniques or tools used to design the applications, the degree of relevance in adopting the stages to the organisation and the different success measures adopted by the companies. In addition, some of the companies participated on the study were not familiar with the terminology used on the methodology.

Therefore, although the confirmatory studies have indicated positive outcomes, the methodology may still need to be validated further to ensure its generality and robustness. Nevertheless, this was minimised by carefully designing the instruments that were used during the confirmatory studies.

8.2.4 Objective 4: Recommendations

The case studies described in this thesis have confirmed the findings from previous research that the success of hypermedia task support systems depends on the usability of such systems, indicated by acceptance from the end users. However, the research has found that the end user usability criteria alone are not sufficient to ensure that the hypermedia systems would continuously be used in the future.

A set of recommendations can therefore be given to those who intend to develop industrial hypermedia task support systems. The recommendations are split into three: for the hypermedia developers (or IT administrators), for the manufacturing task experts (or content providers) and the management.

8.2.4.1 For hypermedia developers

According to the proposed development methodology, the role of hypermedia developers has been shifted from 'creator' to 'facilitator'. To develop a usable and, at the same time, maintainable system, collaboration between the stakeholders of the hypermedia task support systems is required. The developers, therefore, are required to facilitate the creation and management of a collaborative environment.

Nevertheless, the developers are recommended to work in close collaboration with the end users particularly during the design of user interface. More importantly, the developers should understand the task characteristics and how the tasks would be supported by the task support systems.

8.2.4.2 For manufacturing process experts

Within the development cycle of hypermedia task support systems, manufacturing task experts who are required to contribute the content are also considered as the users, who link the *hypermedia developers* with the *end users*.

In Case 3, where a large amount of text was replaced by pictures and video clips, the manufacturing process experts are recommended to carry out task analyses and usability trials during the creation of multimedia objects that support the tasks. Because creating video clips is generally more time consuming than creating still pictures, it is also recommended that the manufacturing process experts work closely with end users in deciding whether pictures and text alone are sufficient, or whether video clips are required.

It has to be borne in mind that direct conversion of printed material into electronic format is not suitable for online presentation. Manufacturing process experts as the content authors are therefore recommended to use online writing techniques as specified in BS 7860 standard. To improve comprehension of the task support document, simple, concise sentences are much preferable than long, complex sentences.

8.2.4.3 For the management

From the management point of view, hypermedia developers, manufacturing process experts and obviously the operators are the internal customers. Therefore, in planning a hypermedia task support development project, managers are recommended to take into account the following issues.

The development process of hypermedia task support systems emphasises collaboration between the hypermedia developers, process experts and the end users. Management should, therefore, provide some training to the manufacturing process experts to support the development of multimedia content. Nonetheless, the type of training would be limited to the use of the collaborative authoring environment provided by the hypermedia developers.

Management should also support the development process by allowing the hypermedia developers and the manufacturing process experts to carry out systems testing and usability evaluation even though this may consume extra resources.

To ensure that a hypermedia task support system is sustainable, management should consider a development team that may consist of people with various expertise

including hypermedia, usability, graphic design and database design. In addition, a project leader is also recommended to manage the progress.

Management should realise that managing a hypermedia project may be different from an ordinary software development project. This is because the hypermedia applications contain graphical design features that do not exist in ordinary software applications. In any hypermedia development project, however, there has always been a risk that the project may take longer to complete than expected, not because of the modifications to the features, but rather due to the look and appearance of the user interface.

8.3 Contribution to knowledge

The contributions made by this research to the body of knowledge are listed below:

1. **New definition of usability**. The research has supported the findings from previous research that the success of creation, implementation and deployment of industrial hypermedia task support systems depends on their usability. Prior to this research, *usability* is defined as a *quality attribute* that assesses how easy the end user can use an IT application [Davis, 1989; Goodhue, 1995] and is often associated with 5 quality components (Figure 23) defined by Nielsen [1995]. This research provides a new definition of usability (see section 6.5) that considers the quality attribute not only from the point of view of the end users, but also from all stakeholders of the application. This will contribute to the Human-Computer Interaction discipline, in which the designers of interactive systems particularly those aimed at an industrial environment, should take into account both the end users and other people that may be involved in the system development.

2. **New measure of usability**. The research also provides two different usability measures (see section 8.1). In the context of hypermedia TSS, the ease with which the end users can access content of the TSS that supports their manufacturing task can be termed *primary usability*, and the ease with which the manufacturing process experts can contribute the content of industrial hypermedia TSS can be termed *secondary usability*. In order to be considered successful, industrial hypermedia task support systems must satisfy both *primary* and *secondary* usability criteria. The research formulates an instrument that can be used to measure those criteria (see section 5.4).

3. **New development methodology**. The research has enhanced many of the techniques to develop industrial hypermedia task support systems (and hypermedia applications in general) found in software/hypermedia engineering literature, such as RMM [Isakowitz *et al*, 1995] and OOHDM [Schwabe and Rossi, 1995]. The unique feature of the *new development methodology* that does not exist in other methodologies, for example, those used in Content

Management Systems (CMS) and other commercial off-the-shelf (COTS) packages, is the design heuristics that aim *to improve an application's coherence* and *to reduce users' cognitive overhead*, two crucial requirements for designing the user interface of industrial task support systems (see section 5.2.2). In addition, the *collaborative authoring environment* that exists within the new development methodology enables the content providers of the industrial hypermedia TSS to carry out task analysis, to create the multimedia objects that are most suitable for the tasks, to author multimedia documents and finally to publish them (Figure 74). By using the collaborative authoring environment, content currency and accuracy can be ensured.

4. **New success model.** The research provides a new success model (Figure 35) that combines TAM [Davis, 1989] and TTF [Goodhue, 1995] with the Software Product Quality model [BSI, 2001]. As with the aforementioned models, the new model is also classified as a predictive model. However, the new model is more comprehensive because it is capable of modelling the two levels of usability. Therefore, the new success model can also be used by researchers in the area of performance improvement, instructional design and training, to evaluate instructional systems including computer-based training, e-learning systems and other conventional, electronic performance support systems (EPSS).

5. **New evidence to support existing belief in multi-skilled project teams.** Within a wider organisational context including the sponsoring company, the research has demonstrated how people from different backgrounds, for example, hypermedia, project management, training, quality management and manufacturing, can form a productive multidisciplinary project team to deliver an industrial hypermedia TSS. Collaborative work between the stakeholders of the application (including hypermedia developers, content providers and the end users) clearly supported this (see section 6.5).

8.4 Limitations of the research

The research has been made possible through sponsorship from Motorola UK, a major provider in global telecommunication solutions. The company has been ideal for conducting experimental work to design, implement and evaluate a number of hypermedia task support systems. The availability of other companies to participate in the confirmatory case studies was also very valuable in the completion of the research. However, the nature of the research and the research settings indicate a number of limitations to the research discussed in the following sections.

8.4.1 Core technology

With the rapidly developed research in hypermedia, web technology and software engineering, particularly during the writing up period of the thesis, a number of new tools and techniques have become mature and are continuously being adopted in commerce and industry. Some of these include the Rational Unified Process (RUP) and the Unified Modelling Language (UML). The latter, which has become a *de-facto* standard in the software development modelling process, is beginning to take over the notations used in conventional hypermedia modelling. UML extensions and adaptations are available to model web/hypermedia applications. The work of Conallen [1998] (finalised by Conallen [2003]) and Ceri *et al* [2003] are examples of such initiatives. In addition, the emergence of WebRatio that is claimed to be the first commercially available CASE tool to generate web pages can potentially make redundant some of the steps in the *formal design*, within the development methodology.

Nevertheless, the development methodology for industrial hypermedia task support systems should not be driven solely by the technology. Instead, it should consider the balance between the state of the art technology and how well the technology will suit industry's needs. For example, as Barry and Lang [2001] found in their survey, although SDLC methodology seems outdated, due to its familiarity and applicability, most multimedia (and hypermedia) developers are still using it in developing hypermedia application particularly in general industry.

The stages in the proposed development methodology, to a large extent, share similarities with the phases outlined in both RUP and the spiral model. The underlying principle is the never ending iteration of requirements identification, analysis/design, implementation and evaluation. The ultimate goals are to ensure that the product has met the requirements and to minimise the risk of failure.

8.4.2 Validation of the methodology

During the second part of the research execution, the turbulence in the telecommunication industry had hit the sponsoring company somewhat unexpectedly. The company had made many of the shop floor workers redundant, including some of the key persons in the project. This resulted in slow progress and problems in conducting further case studies to further validate the development methodology within the sponsoring company.

For example, since no specific questionnaire items to measure maintainability could be located from the literature, these were then formulated based on the dimensions adopted from ISO/IEC 9126 [BSI, 2001]. Further testing is necessary in order to ensure that the items exhibit validity and reliability. With the dramatic changes in the sponsoring company, it was not possible to conduct further experiments and to gather quantitative

data, because of the limited availability of technicians (the manufacturing process experts).

Further limitations were due to the number of companies that have implemented task support systems using hypermedia technology. Among those companies approached, only four wished to participate in the confirmatory case studies. Ideally, the confirmatory case studies would have been further expanded to incorporate more companies, particularly manufacturing organisations.

8.4.3 Measurement of success

Research in the area of Information Systems has been trying to understand the link between the use of information systems and organisational performance, in order to justify the success of such systems. However, despite the vast amount of work carried out, the objective measures of systems success are difficult to identify [Goodhue, 1995]. Researchers have used the user evaluations of systems as a surrogate measure. The most common criteria used to indicate the success were the system's *usage* [Davis, 1989] and the degree of fit between the task and the information system to support the task or *task-technology fit* [Goodhue, 1995].

For that reason, the proposed success model that was used to evaluate the task support systems did not intend to measure the direct impact of the task support systems in terms of task performance. Instead, the *intention to use* was employed to indicate whether or not the systems will be used and will continue to be used by the users. This research has considered the findings from previous researchers (for example Davis [1989], DeLone and McLean [1992] and Goodhue [1995]) who argued that the system's usage is a pre-requisite that needs to be fulfilled before the task performance.

8.5 Future work

As emphasised by Lowe and Hall [1999], the development methodology for hypermedia will continue to evolve with the advancement of Internet technology and software engineering. Awareness of state of the art research in those areas would therefore be valuable to improve the proposed development methodology. New tools and approaches might help further research to continuously refine the methodology, allowing industrialists to develop task support systems in a more effective and efficient way.

Future work might focus on carrying out more experiments to further validate the development methodology. Such work should consider the applicability of the methodology within the sponsoring company by conducting more case studies to validate the maintainability construct. Survey and confirmatory studies of other industries that have developed task support systems using hypermedia technology might have made the methodology more robust.

Further research is required to devise the methods to quantify the benefits that might accrue from implementing industrial hypermedia applications. One question that can be raised against the implementation of such systems is whether the benefits actually result from the task support systems and are not caused by other improvement initiatives, such as quality programmes and lean manufacturing.

Finally, further research should investigate the extent to which industrial hypermedia systems actually support job performance, for example, by investigating the impact of the availability of the task support system to the production rate or productivity. Another method may be to compare the effectiveness of the hypermedia task support system with other types of task support system, including paper-based TSS and traditional training. Since task support systems are a means of delivering knowledge and hence a type of training method, further research may adopt methods to evaluate training effectiveness, for example level 4 of Kirkpatrick's method [Kirkpatrick, 1998].

References

1. Akao, Y. (1990). Quality Function Deployment: Integrating Customer Requirements into Product Design, Productivity Press

2. Balasubramanian, V. and Bashian, A. (1998). Document Management and Web Technologies: Alice Marries the Mad Hatter, Communications of the ACM, Vol. 41, No. 7, pp. 107-115

3. Banerji, A. (1999). Performance Support in Perspective, Performance Improvement, Vol. 38, No. 7, pp. 6-9

4. Barnett, W. D. and Raja, M. K. (1995). Application of QFD to the software development process, International Journal of Quality & Reliability Management, Vol. 12, No. 6, pp. 24-42

5. Barry, C. and Lang, M. (2001). A Survey of Multimedia and Web Development Techniques and Methodology Usage, IEEE Multimedia, April-June, pp. 52-60

6. Baskerville, R. L. (1999). Investigating Information Systems with Action Research, Communications of the Association of Information Systems, 2, No. 19

7. Bell, J. (1987). Doing Your Research Project A Guide for First-Time Researchers in Education and Social Science, 2nd Ed., Open University Press, Buckingham

8. Bernard, M., Liao, C. H., and Mills, M. (2001). The effects of font type and size on the legibility and reading time of online text by older adults, Proceedings of the CHI'2001 Conference on Human Factors in Computing Systems, Seattle, Washington

9. Bieber, M., Vitali, F., Ashman, H., Balasubramanian, V., and Oinas-Kukkonen, H. (1997). Some Hypermedia Ideas for the WWW, Proceedings of the 30th Hawai International Conference on System Sciences, pp. 309-319

10. Boehm, B (1988). A Spiral Model of Software Development and Enhancement, IEEE Computer, Vol. 21, No. 5, pp. 61-72

11. Booch, G. (1994). Object-oriented Design with Applications, 2nd Ed, Benjamin Cummings, Redwood City, California

12. Booch, G., Rumbaugh, J., and Jacobson, I. (1998). The Unified Modeling Language User Guide, Addison-Wesley

13. Brereton, P., Budgen, D., and Hamilton, G. (1998). Hypertext: The Next Maintenance Mountain, Computer, December, pp. 49-54

14. Britton, C., Jones, S., Myers, M., and Sharif, M. (1997). A survey of current practice in the development of multimedia systems, Information and Software Technology, Vol. 39, pp. 695-705

15. Brusilovsky, P. and Cooper, D. W. (1999). ADAPTS: Adaptive Hypermedia for a Web-Based Performance Support System, Proceedings of the 2nd Workshop on Adaptive Systems and User Modeling on the WWW

16. Brusilovsky, P. and Cooper, D. W. (2002). Domain, Task and User Models for an Adaptive Hypermedia Performance Support System, Proceedings of the 2002 International Conference on Intelligent User Interfaces, ACM Press, San Francisco, pp. 23-30

17. Bryman, A. (1989). Research Methods and Organization Studies, Routledge, Taylor and Francis Group, London

18. BSI (1995). BS ISO 10013: 1995. Guidelines for developing Quality Manuals

19. BSI (1996). BS 7830:1996. Guide to the design and preparation of on-screen documentation for users of application software

20. BSI (2001). BS ISO/IEC 9126-1:2001. Information technology. Software product quality. Quality model

21. Bullock, D. H. (1982). Guiding Job Performance with Job Aids, Training and Development Journal, September, pp. 36-40

22. Campbell, C. P. (1996). Job Performance Aids, Journal of European Industrial Training, Vo. 20, No. 6, pp. 3-21

23. Carr, C. (1992). PSS! Help when you need it, Training & Development, June, pp. 31-38

24. Ceri, S., Fraternali, P., and Matera, M. (2002). Conceptual Modelling of Data-intensive Web Applications, IEEE Internet Computing, Vol. 6, No. 4, pp. 20-30

25. Ceri, S., Fraternali, P., Bongio, A., Brambilla, M., Comai, S., and Matera, M. (2003). Designing Data-intensive Web Applications, Morgan Kauffmann Publishers, San Francisco, CA

26. Chan, A., Marks, S., Mattey, D., Wu, B., and Tjahjono, B. (2000). ManTIS - Manufacturing Test Information System, Proceedings the Hermes 2000 Symposium, Ft. Lauderdale, FL, USA

27. Checkland, P. (1981). Systems Thinking, Systems Practice, John Wiley & Sons, Chichester, England

28. Chen, P. (1976). The Entity-Relationship Model: Towards a Unified View of Data, ACM Transactions on Database Systems, Vol. 1, No. 1, pp. 1-36

29. Clark, R. C. (1992). EPSS-Look before you leap: Some cautions about applications of Electronic Performance Support Systems, Performance & Instructions, May/June, pp. 22-25

30. Coda, F., Ghezzi, C., Vigna, G., and Garzotto, F. (1998). Towards a Software Engineering Approach to Web Site Development, Proceedings of the 9th IEEE International Workshop on Software Specification and Design (IWSSD), Ise-Shima (Isobe), Japan

31. Cole, K., Fischer, O., and Saltzman, P. (1997). Just-in-Time Knowledge Delivery, Communications of the ACM, Vol. 40, No. 7, pp. 49-53

32. Conallen, J. (1998). Modeling Web Application Architectures with UML, Communications of the ACM, Vol. 42, No. 10, pp. 62-70

33. Conallen, J. (2003). Building Web Applications with UML, Addison-Wesley, Boston, MA

34. Coughlan, P. and Coghlan, D. (2002). Action Research for Operations Management, International Journal of Operations & Production Management, Vol. 22, No. 2, pp. 220-240

35. Cowan, D. D. and Lucena, C. J. P. (1995). Abstract Data Views: An Interface Specification Concept to Enhance Design for Reuse, IEEE Transactions on Software Engineering, Vol. 21, No. 3, pp. 229-243

36. Crowder, R. M., Hall, W., Heath, I., and Wills, G. (1999). Integration of Manufacturing Information Using Open Hypermedia, Computers in Industry, Vol. 38, pp. 31-42

37. Crowder, R. M., Hall, W., Heath, I., and Wills, G. (2000). Industrial Strength Hypermedia: Design, Implementation and Application, International Journal of Computer Integrated Manufacturing, Vol. 13, No. 3, pp. 173-186

38. Crowder, R., Wills, G., and Hall, W. (2003). Evaluation of a hypermedia maintenance support application, Computers in Industry, Vol. 51, pp. 327-344

39. Davis, F. D. (1989). Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology, MIS Quarterly, September, pp. 319-339

40. Davis, F. D., Bagozzi, R. P., and Warshaw, P. R. (1989). User acceptance of computer technology: a comparison of two theoretical models, Management Science, Vol. 35, No. 8, pp. 982-1003

41. DeLone, W. H. and McLean, E. R. (1992). Information systems success: the quest for the dependent variable, Information systems research, Vol. 3, No. 1, pp. 60-95

42. Diaz, A. and Isakowitz, T. (1995). RMCase: Computer-Aided Support for Hypermedia Design and Development, International Workshop on Hypermedia Design, Montpellier, France

43. Dickelman G. J. (2000). Performance Support in Internet Time: The State of the Practice, Performance Improvement, Vol. 39, No. 6, pp. 7-17

44. Dishaw, M. T. and Strong, D. M. (1998). Assessing Software Maintenance Tool Utilization using Task-technology Fit and Fitness-for-use Models, Software Maintenance: Research and Practice, Vol. 10, pp. 151-179

45. Duke-Moran, C., Swope, G., Morariu, J., and deKam, P. (1999). Performance Support Case Studies from IBM, Performance Improvement, Vol. 38, No. 7, pp. 28-32

46. Elsenheimer, J. (1998). Job Aids in the Technology Age, Performance Improvement, Vol. 37, No. 8, pp. 32-35

47. Fakun, D. (2000). The Development of Successful Industrial Hypermedia Applications, PhD Thesis, Cranfield University, Cranfield, UK

48. Fakun, D. and Greenough, R. M. (2002). User-interface design heuristics for developing usable industrial hypermedia applications, Human Factors and Ergonomics in Manufacturing, Vol. 12, No. 2, pp. 127-149

49. Faulkner, X. (2000). Usability Engineering, McMillan, Basingstoke, UK

50. Frieling, E., Freiboth, M., Henniges, D., and Saager, C. (1997). Effects of team work on the working conditions of short cycled track work: A case study from the European automobile industry, International Journal of Industrial Ergonomics, 20, pp. 371-388

51. Garzotto, F., Mainetti, L., and Paolini, P. (1995). Hypermedia Design, Analysis and Evaluation Issues, Communications of the ACM, Vol. 38, No. 8, pp. 74-86

52. Garzotto, F., Paolini, P., and Schwabe, D. (1993). HDM - A Model-based Approach to Hypertext Application Design, ACM Transactions on Information Systems, Vol. 11, No. 1, pp. 1-26

53. Gery, G. (1991). Electronic Performance Support System: How and why to remake the workplace through the strategic application of technology, Weingarten Publications, Boston, MA

54. Gery, G. (1995). Attributes and Behaviors of Performance-Centered Systems, Performance Improvement Quarterly, Vol. 8, No. 1, pp. 47-93

55. Ginige, A., Lowe, D. B., and Robertson, J. (1995). Hypermedia Authoring, IEEE Multimedia, Vol. 2, No. 4, pp. 24-35

56. Ginige, A. and Murugesan, S. (2001). Web Engineering: An Introduction, IEEE Multimedia, January-March, pp. 14-18

57. Goodhue, D. L. (1995). Understanding User Evaluations of Information Systems, Management Science, Vol. 41, No. 12, pp. 1827-1844

58. Goodhue, D. L. and Thompson, R.L. (1995). Task-Technology Fit and individual performance, MIS Quarterly, Vol. 19, No. 2, pp.213-236

59. Goodhue, D. L. (1998). Development and Measurement Validity of a Task-Technology Fit Instrument for User Evaluations of Information Systems, Decision Sciences, Vol. 29, No. 1, pp. 105-138

60. Greenough, R. M., Fakun, D., and Kay, J. M. (2000). An exploratory study of the application of digital documents to light manual assembly, Proceedings of the 7th International Conference on Human Aspects of Advanced Manufacturing: Agility & Hybrid Automation - HAAMAHA 2000, Krakow, Poland, pp. 37-40

61. Greenough, R. M., Fakun, D., and Kay, J. M. (2001). Development of a digital manual for a manufacturing system - a case study, Integrated Manufacturing Systems, Vol. 12, No. 6, pp. 387-394

62. Gunasekaran, A. and Love, P. E. D. (1999). A Review of Multimedia Technology in Manufacturing, Computers in Industry, Vol. 38, pp. 65-76

63. Gygi, K. (1991). Recognising the symptoms of hypertext ... and what to do about it, In Laurel, B. (Ed), The art of human computer interface design, Addison-Wesley, Reading, MA, pp. 279-287

64. Hackos, J. and Redish, J. (1998). User and Task Analysis for Interface Design, John Wiley & Sons

65. Halasz, F. and Schwartz, M. (1994). The Dexter hypertext reference model, Communications of the ACM, Vol. 37, No. 2, pp. 30-39

66. Hall, B. (1998). Web-based Training Cookbook, John Wiley & Sons, USA

67. Hall, W. (1995). The Role of Hypermedia in Multimedia Information Systems, ACM Computing Surveys, Vol. 27, No. 4, pp. 599-601

68. Hardman, L. (1995). Experiences in authoring hypermedia: creating better presentations, In Schuler, W., Hannemann, J. and Streitz, N. (Eds), Research Reports ESPRIT Project 6532 HIFI Volume 1, Springer-Verlag, pp. 16-26

69. Hasebrook, J. (1999). Web-based Training, Performance and Controlling, Journal of Network and Computer Applications, Vol. 22, pp. 51-64

70. Haywood, R. (1999). From Paper to Intranet, IIE Solutions, February, pp. 47-49

71. Hernandez, M. J. and Viescas, J. L. (2000). SQL Queries for Mere Mortals, Addison-Wesley

72. Hosni, Y. A. and Okrasi, A. E. (1992). Development procedure of a hypermedia application in support of industrial tasks, Computers and Industrial Engineering, Vol. 23, Nos. 1-4, pp. 313-317

73. Hulme, C. (1984). Extracting Information from Printed and Electronically Presented Text, In Monk, A. (Ed.), Fundamentals of Human-Computer Interaction, Academic Press, London, UK

74. Ince, M. and Kay, J. M. (1998). The development of a user-oriented information systems development strategy, Proceedings of the 14th National Conference on Manufacturing Research, pp. 481-486

75. Isakowitz, T., Stohr, E., and Balasubramanian, P. (1995). RMM: a Methodology for Structured Hypermedia Design, Communications of the ACM, Vol. 38, No. 8, pp. 34-44

76. Jacobson, I., Booch, G., and Rumbaugh, J. (1999). The Unified Software Development Process, Addison-Wesley, Reading, MA

77. Jiang, J J., Klein, G., and Balloun, J. (1998). Perceptions of system development failures, Information and software technology, Vol. 39, pp. 933-937

78. Kasvi, J. J. J., Pulkkis, A., Vartiainen, M., and Nieminen, M. (1994). Building an Information System to Support Shop-Floor Employees, Proceedings of the Interdisciplinary Workshop on Complex Learning in Computer Environments, pp. 186-190

79. Kasvi, J. J. J., Vartiainen, M., Pulkkis, A., and Nieminen, M. (1996). Information tools for the shop floor, AI & Society, Vol. 10, pp. 26-38

80. Kasvi, J. J. J., Vartiainen, M., Pulkkis, A., and Nieminen, M. (2000). The Role of Information Support Systems in the Joint Optimization of Work Systems, Human Factors and Ergonomics in Manufacturing, Vol. 10, No. 2, pp. 193-221

81. Kennedy, W. J., Gramopadhye, A. K., and Ramu, N. V. (1995). Hypertext in A Manufacturing Environment, Production and Inventory Management Journal, Second Quarter, pp. 4-7

82. Kirkpatrick, D. L. (1998). Evaluating Training Programs: the Four Levels, Berrett-Koehler Publishers, Inc., San Francisco

83. Kirwan, B. and Ainsworth, L. K. (1992). A Guide to Task Analysis, Taylor & Francis, London

84. Koshy, T. T., Gramopadhye, A. K., Kennedy, W. J., and Ramu, N. V. (1996). Application of Hypertext Technology to Assist Maintenance on the Shop Floor, Computers and Industrial Engineering, Vol. 30, No. 2, pp. 283-295

85. Kroll, P. and Kruchten, P. (2003). The Rational Unified Process made easy: a practitioner's guide to the RUP, Addison-Wesley

86. Laffey, J. (1995). Dynamism in Electronic Performance Support Systems, Performance Improvement Quarterly, Vol. 8, No. 1, pp. 31-46

87. Lang, M. (2001). A Study of Practice in Hypermedia Systems Design, Doctoral Consortium, European Conference on Information Systems (ECIS), Slovenia

88. Lang, M. and Barry, C. (2001). Techniques and Methodologies for Multimedia Systems Development: A Survey of Industrial Practice, In: Russo, N. L. et al (Eds), Realigning Research and Practice in Information Systems Development: the Social and Organisational Perspective, Kluwer, Boston, pp. 77-86

89. Lange, D. B. (1994). An Object-Oriented Design Method for Hypermedia Information Systems, Proceedings of the 27th Annual Hawaii International Conference on System Sciences (HICSS '94), IEEE Computer Society Press, pp. 366-375

90. Leat, M. (1998). Human Resource Issues of the European Union, Financial Times Management, London

91. Lee, H., Kim, J., Kim, Y. G., and Cho, S. H. (1999(a)). A View-Based Hypermedia Design Methodology, Journal of Database Management, Vol. 10, No. 2, pp. 3-13

92. Lee, H., Lee, C., and Yoo, C (1999(b)). A scenario-based object-oriented hypermedia design methodology, Information & Management, Vol. 36, pp. 121-138

93. Leung, R., Leung, H., and Hill, J. (1995). Distributed Quality Manual via Hypermedia, Proceedings of the 2nd International Conference on Industrial Automation, Nancy, France

94. Lowe, D. and Hall, W. (1999). Hypermedia & the Web: An Engineering Approach, John Wiley & Sons Ltd, Chichester, West Sussex, England

95. Lucas, H. C. Jr. (1978). Empirical Evidence for a Descriptive Model of Implementation, MIS Quarterly, Vol. 2, No. 2, pp. 27-42

96. MacMillan, R. H. (1996). Moving to a Paperless Quality Management System: Aardvark to Zulu, British Telecommunications Engineering, Vol. 15, pp. 100-104

97. MacMillan, R. H. (1997). Aardvark to Zulu, Software Quality Journal, Vol. 6, pp. 151-156

98. Malcolm, K. C., Poltrock, S. E., and Schuler, D. (1991). Industrial Strength Hypermedia: Requirements for a Large Engineering Enterprise, Proceedings of Hypertext '91, December, pp. 13-24

99. Marchionini, G. and Schneiderman, B. (1988). Finding facts versus browsing knowledge in hypertext systems, IEEE computer, Vol. 21, No. 1, pp. 70-80

100. Marcus, A., Smilonich, N., and Thompson, L. (1995). The cross-GUI handbook for multiplatform user interface design, Addison-Wesley, Reading

101. Maskell, B. H. (1994). Software and the Agile Manufacturer, Productivity Press, Portland, Oregon

102. Mathieson, K. and Keil, M. (1998). Beyond the Interface: Ease of Use and Task-technology Fit, Information & Management, Vol. 34, pp. 221-230

103. Mayer, R. E. and Anderson, R. B. (1991). Animations Need Narations: An Experimental Test of a Dual-Coding Hypothesis, Journal of Educational Psychology, Vol. 83, No. 4, pp. 484-490

104. Mayer, R. E. and Gallini, J. K. (1990). When Is an Illustration Worth Ten Thousand Words?, Journal of Educational Psychology, Vol. 82, No. 4, pp. 715-726

105. McGraw, K. L. (1994). Performance Support Systems: Integrating AI, Hypermedia, and CBT to Enhance User Performance, Jl. of Artificial Intelligence in Education, Vol. 5, No. 1, pp. 3-26

106. Monden, Y. (1994). Toyota Production System: an integrated approach to just-in-time, 2nd Ed., Chapman & Hall, London, UK

107. Mosley, D. J. (1993). The Handbook of MIS Application Software Testing: Methods, Techniques, and Tools for Assuring Quality through Testing, Prentice-Hall

108. Murugesan, S., Deshpande, Y., Hansen, S., and Ginige, A. (2001). Web Engineering: A New Discipline for Development of Web-Based Systems, In Murugesan, S. & Deshpande, Y. (Eds), Web Engineering: Managing Diversity and Complexity of Web Application Development, pp. 3-13

109. Najjar, L. J. (1995). Does multimedia help people learning? Technical Report GIT-GVU-95-28, Georgia Institute of Technology

110. Nanard, J. and Nanard, M. (1995). Hypertext Design Environments and the Hypertext Design Process, Communications of the ACM, Vol. 38, No. 8, pp. 49-56

111. Nelson, T. H. (1995). The Heart of Connection: Hypermedia Unified Transclusion, Communications of the ACM, Vol. 38, No. 8, pp. 31-33

112. Neuman, W. L. (1997). Social Research Methods: Qualitative and quantitative approaches, 3rd Ed, Allyn and Bacon, Boston, London

113. Nielsen, J. (1995). Multimedia and Hypertext: The Internet and Beyond, Academic Press, Inc., California

114. Nieminen, M., Kasvi, J. J. J., Pulkkis, A., and Vartiainen, M. (1995). Interactive task support on the shop floor: Observations on the usability of the interactive task support system and differences in orientation and hands-on training use, In Kirby, M.A.R., Dix, A.J. and Finlay, J.E. (Eds.), People and Computers X, Cambridge University Press, Cambridge, England, pp. 7993

115. Oakland, J. (1993). Total Quality Management, 2nd Ed., Butterworth-Heinemann Ltd., Oxford, England

116. Ockerman, J. J., Najjar, L. J., and Thompson, C. (1999). FAST: Future Techonology for Today's Industry, Computers in Industry, Vol. 38, pp. 53-64

117. Olsina, L. (1998). Building a Web-based information system applying the Hypermedia Flexible Process Modeling Strategy, Proceedings of the 1st International Workshop on Hypermedia Development, Hypertext'98

118. Pham, D. T., Dimov, S. S., and Setchi, R. M. (1999). Intelligent Product Manuals, Proc Instn Mech Engrs, Vol. 213, No. 1, pp. 65-76

119. Powell, T. A., Jones, D. L., and Cutts, D. C. (1998). Web Site Engineering - Beyond Web Page Design, Prentice-Hall, Upper Saddle River, NJ, USA

120. Raybould, B. (1995). Performance Support Engineering: An Emerging Development Methodology for Enabling Organizational Learning, Performance Improvement Quarterly, Vol. 8, No. 1, pp. 7-22

121. Raybould, B. (2000). Building Performance-Centered Web-Based Systems, Information Systems, And Knowledge Management Systems In the 21st Century, Performance Improvement, Vol. 39, No. 6, pp. 32-39

122. Rea, H. J., Falconer, R. A., Murray, J. L., and Simmons, J. E. L. (1998). Adaptive Instructions for Assembly, Proceeding of the 3rd International Conference on Managing Innovative Manufacturing (MIM 98), pp. 247-252

123. Rettig, M. (1994). Prototyping for tiny fingers, Communications of the ACM, Vol. 37, No. 4, pp. 21-27

124. Reynolds, A. and Araya, R. (1995). Building Multimedia Performance Support Systems, McGraw Hill

125. Robson, C. (1993). Real World Research: A Resource for Social Scientist and Practitioner Researcher, Blackwell Publishers Ltd, Oxford, UK

126. Rosenberg, M. J. (1995). Performance Technology, Performance Support and the Future of Training: a commentary , Performance Improvement Quarterly, Vol. 8, No. 1, pp. 94-99

127. Royce, W. (1970). Managing the development of large software systems: concepts and techniques. In Proc. IEEE WESTCON, pp. 1-9

128. Schwabe, D., Pontes, R. A., and Moura, I. (1999). OOHDM-Web: An Environment for Implementation of Hypermedia Applications in the WWW, ACM SigWEB Newsletter, Vol. 8, No. 2

129. Schwabe, D. and Rossi, G. (1995). The Object-Oriented Hypermedia Design Model, Communications of the ACM, Vol. 38, No. 8, pp. 45-46

130. Shadur, M. A., Rodwell, J. J., and Bamber, G. J. (1995). Factors predicting employees' approval of lean production, Human Relations, Vol. 48, No. 12, pp. 1403-1425

131. Shneiderman, B. (1997). Designing information abundant web sites: issues and recommendations, International journal of human computer studies, Vol. 47, pp. 5-29

132. Silva, A., Andrade, G., and Delgado, J. (1996). A Multimedia Database Supporting a Generic Computer Based Quality Management System, The 9th ERCIM Database Research Group Workshop on Multimedia Database Systems, Darmstadt, Germany,

133. Stevens, G. H. and Stevens, E. F. (1995). Designing Electronic Performance Support Tools, Educational Technology Publications, Inc., Englewood Cliffs, New Jersey,

134. Stolovitch, H. D. (2000). Human Performance Technology: Research and Theory to Practice, Performance Improvement, Vol. 39, No. 4, pp. 7-16

135. Suh, W. and Lee, H. (2001). A methodology for building content-oriented hypermedia systems, Journal of systems and software, Vol. 56, pp. 115-131

136. Suzaki, K. (1993). The New Shop Floor Management: Empowering People for Continuous Improvement, Free Press, New York

137. Tan, K. C., Xie, M., and Chia, E. (1998). Quality Function Deployment and Its Use in Designing Information Technology Systems, International Journal of Quality & Reliability Management, Vol. 15, No. 6, pp. 634-645

138. Taylor, M., McWilliam, J., Sheehan, J., and Mulhaney, A. (2002). Maintenance issues in the Web site development process, Journal of Software Maintenance and Evolution: Research and Practice, Vol. 14, No. 2 , pp. 109 - 122

139. Thüring, M., Hannemann, J., and Haake, J. M. (1995). Hypermedia and Cognition: Designing for Comprehension, Communications of the ACM, Vol. 38, No. 8, pp. 57-66

140. Tjahjono, B. and Greenough, R. M. (2002). Using collaborative authoring to develop a hypermedia performance support system, Performance Improvement, Vol. 41, No. 6, pp. 34-41

141. Tullis, T. S., Boynton, J. L., and Hersh, H. (1995). Readability of fonts in the Windows environment, Proceedings of the Conference on Human Factors in Computing Systems (CHI'95), Denver, Colorado

142. Vartiainen, M., Pulkkis, A., Kasvi, J. J. J., and Nieminen, M. (1996). The Critical Job and Organizational Demands for the Design of Information Support Systems, In: Hacker W., Richter, P. & Wagner, T. (Eds.) Analysis and Design on Mental Work, pp. 19-36

143. Venkatesh, V. and Davis, F. D. (1996). A Model of the antecedents of perceived ease of use: a development and test, Decision Sciences, Vol. 27, No. 3, pp. 451-481

144. Wu, B., Tjahjono, B., and Looks, V. (1999). An Intranet-based Multimedia Quality Management System, Proceedings of the 15th International Conference on Production Research (ICPR-15), Limerick, Ireland, pp. 1103-1106

145. Yin, R. K. (1994). Case Study Research - Design and Methods, Sage Publications

146. Zairi, M. and Youssef, M. A. (1995). Quality function deployment: a main pillar for successful total quality management and product development, International Journal of Quality & Reliability Management, Vol. 12, No. 6, pp. 9-23

147. Zigurs, I., Buckland, B. K., Connolly, J. R., and Wilson, E. V. (1999). A Test of Task-technology Fit Theory for Group Support Systems, The DATA BASE for Advances in Information Systems, Vol. 30, No. 3, pp. 34-50

APPENDIX A

HDM notations

HDM uses of a number of modelling elements, known as *primitives*, mostly adopted from both the Dexter model [Halasz and Schwartz, 1994] and the E-R model [Chen, 1976]. They are *entities*, *components*, *perspectives*, *units* and *links*. In addition, a *schema* is used to specify the structure of the hypermedia application. HDM also employs another primitive called *outline* to describe the access structure and to locate the most convenient access point at which to start.

Figure A-1 shows a simple HDM schema to illustrate the relationship between HDM primitives connected by the three links. *Entity* is the fundamental primitive in HDM, representing a physical or conceptual object of the information domain. An example of an entity in a library information system is 'book', 'author' or 'lender'. An entity is made up of one or more *components*. Components inherit their type from their entity and can only exist as part of an entity. An example of a component is 'biography' or 'list of publications' that is part of the 'author' entity. In hypertext application, it is often desirable to present the same topic in different ways, for instance according to the user's preference. Perspective is used to represent the different presentations of the same entity or component. An example would be that the 'biography', which is a component of the 'author' entity, may be presented in two different versions using two different perspectives: 'summary' and 'detail'. A *unit* is associated with a specific perspective. In more generic hypertext terminology, a unit is also known as a *node*. Examples are 'summary_in_English' and 'summary_in_French'.

The HDM information structure can be interconnected by *links*. HDM acknowledges three types of links: structural, perspective and application. Structural links connect together components belonging to the same entity. An example of a structural link is the link from 'biography' to 'list of publications'. Perspective links connect the different units corresponding to the same component, for example the link between 'summary' and 'detail' biography. Finally, application links are used to describe non-structural relationships between different entities, for instance the link between the author and the book the author wrote.

A *schema* in HDM is a specification of a hypermedia application in terms of entity types and link types. It is also derived from database fields describing the file structures. An *outline* is used as an entry point to the hypermedia application before traversing along the link to browse the entities or components. An outline can also be described as a hierarchy of components within an entity.

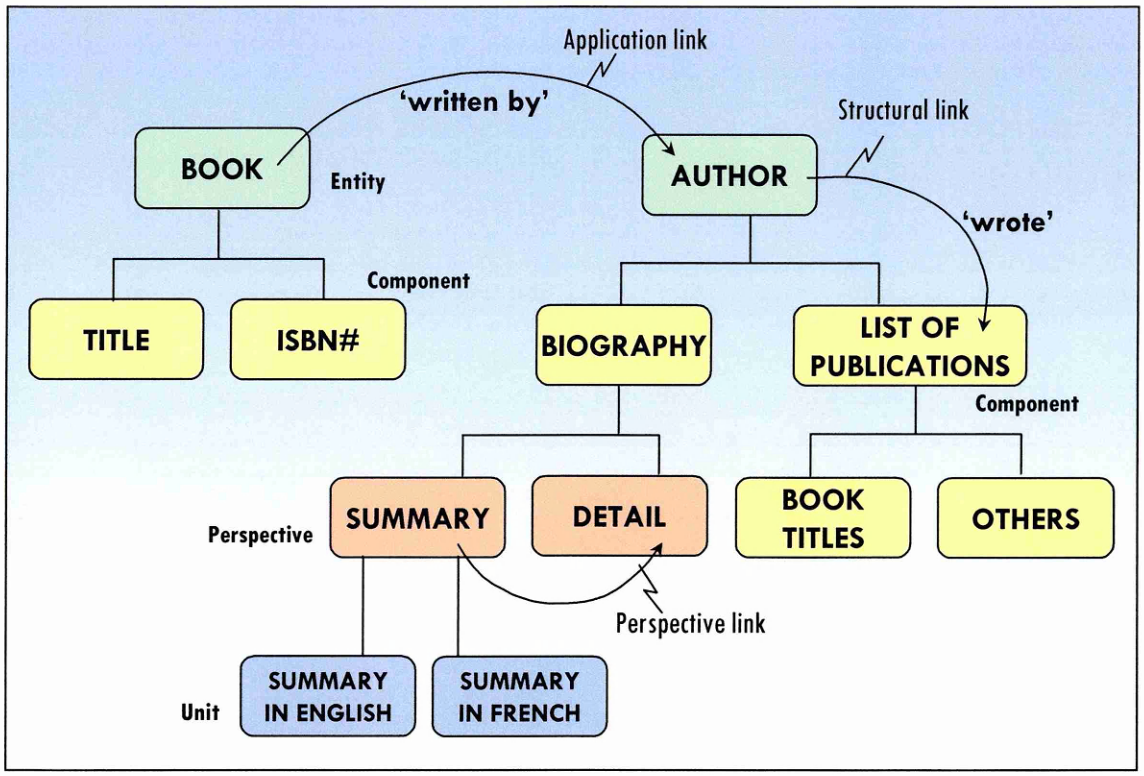


Figure A-1 – HDM primitives

Note that no prescribed symbols are given in HDM, and hence the notations in the above figure are intended only as an aid to understanding the HDM primitives. Looking at the primitives used, HDM shares some apparent similarities with Entity Relationship (E-R) model [Chen, 1976], particularly because the term *entity* was adopted from E-R model. However, Garzotto *et al* [1993] argue that HDM entities are more structured than the E-R entities, which do not have structural links.

RMM notations

RMM's data model (known as RMDM) [Isakowitz *et al*, 1995] uses modelling elements or notations. These symbols provide a language for describing the information objects and navigation mechanism in hypermedia application. RMDM elements (Figure A-2) are based on both HDM and E-R for describing the information together with navigation symbols.

The *index* acts as a table of contents, providing a direct link to each listed entity A-3(a). A *guided tour* provides a linear access through a collection of entities, allowing the user to move forwards or backwards on a path Figure A-3(b). A combination of index and guided tour is called an *indexed guided tour* and allows a user to access the table of contents and traverse through each entity listed on the index Figure A-3(c). *Grouping* is a menu-like mechanism that serves as an entry point to the hypermedia application. A typical example is an opening menu where a user will have an option to access other features, such as indices and guided tours.

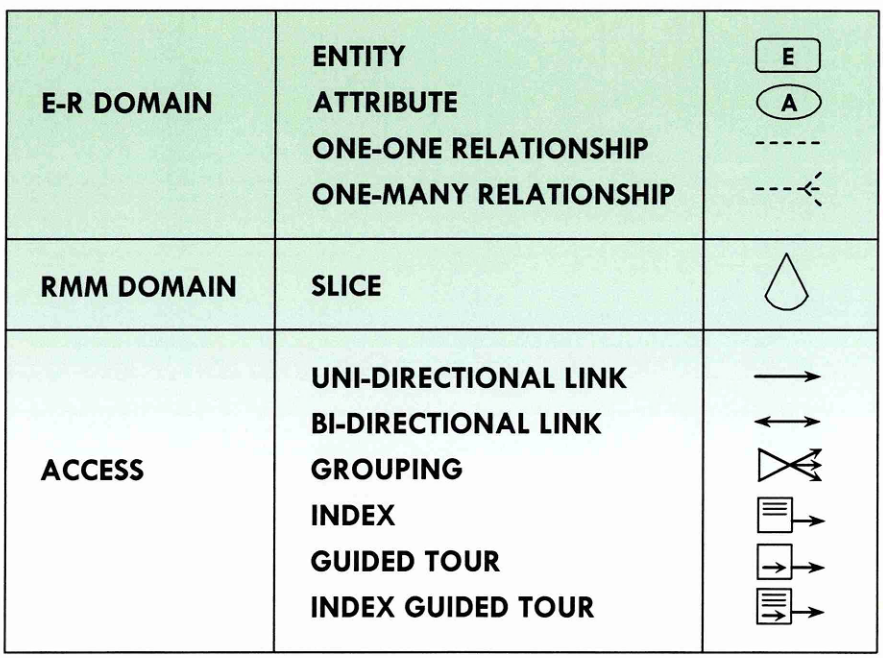


Figure A-2 – RMDM elements [Isakowitz *et al*, 1995]

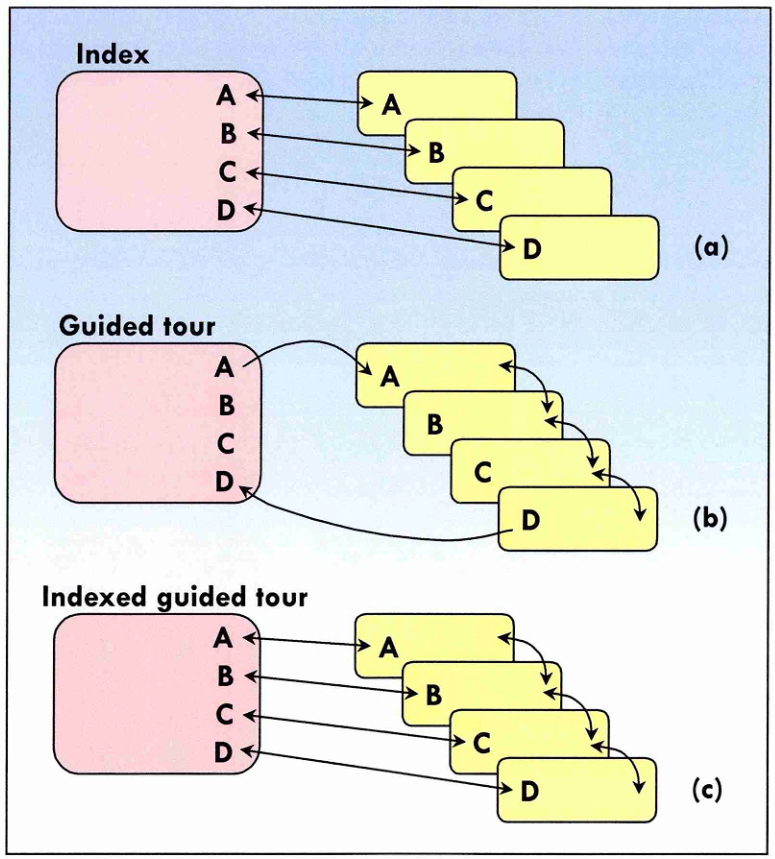


Figure A-3 - RMDM access path [Isakowitz *et al*, 1995]

APPENDIX B

ManTIS simulation mode – Questionnaire for Operators

Table B-1 – Items chosen to collect task familiarity

		Strongly disagree						*Strongly agree*
TC01	I am familiar with *Horizonmacro* product	1	2	3	4	5	6	7
TC02	I am familiar with Initialisation and Alarm test, and RF test procedures	1	2	3	4	5	6	7

Table B-2 – Items chosen to measure perceived ease of use (EOU)

		Strongly disagree						*Strongly agree*
EOU01	Learning to use the program is easy for me	1	2	3	4	5	6	7
EOU02	I find it takes a lot of effort to become skilful at using the program*	1	2	3	4	5	6	7
EOU03	Interacting with the program is frustrating for me*	1	2	3	4	5	6	7
EOU04	Overall, I find the program easy to use	1	2	3	4	5	6	7

* = reverse-coded items

Table B-3 – Items chosen to measure perceived usefulness (U)

		Strongly disagree						*Strongly agree*
U01	I believe the program will make my job easier to complete	1	2	3	4	5	6	7
U02	I think the program will support me to do my job	1	2	3	4	5	6	7
U03	I think the program will improve my job performance	1	2	3	4	5	6	7
U04	Overall, I think the program is useful	1	2	3	4	5	6	7

Table B-4 – Items chosen to measure perceived task-technology fit (TTF)

		Strongly disagree						*Strongly agree*
CUR02	The information from the program is up to date	1	2	3	4	5	6	7
MEA01	I can understand easily the definition or the meaning of the data from the program	1	2	3	4	5	6	7
ACC02	It is easy to access the program and to get the information that I want	1	2	3	4	5	6	7
ACU01	The information on the program is accurate enough for my purposes	1	2	3	4	5	6	7
REL02	The program is subject to frequent hardware and software problems and crashes	1	2	3	4	5	6	7
ACC01	I can easily get the information to support my task from the program	1	2	3	4	5	6	7
ACU02	The are a number of errors and inaccurate information on the program	1	2	3	4	5	6	7
MEA02	The exact meaning of the information I found from the program is obvious	1	2	3	4	5	6	7
CUR01	I cannot get the information from the program that is current enough for my purpose	1	2	3	4	5	6	7
REL01	I can count on the program to be up and available when I need it	1	2	3	4	5	6	7

Table B-5 – Items chosen to measure intention to use

		Strongly disagree						*Strongly agree*
IU01	I intend to use the program if I have problems	1	2	3	4	5	6	7
IU02	I would use and recommend the program to my colleague	1	2	3	4	5	6	7

ManTIS reference and training modes - Questionnaire for Operators

Table B-6 – Items chosen to measure Task-Technology Fit (TTF)

		Strongly disagree						*Strongly agree*
CUR02	The information from the program is up to date	1	2	3	4	5	6	7
MEA01	I can understand easily the definition or the meaning of the data from the program	1	2	3	4	5	6	7
ACC02	It is easy to access the program and to get the information that I want	1	2	3	4	5	6	7
ACU01	The information on the program is accurate enough for my purposes	1	2	3	4	5	6	7
REL02	The program is subject to frequent hardware and software problems and crashes	1	2	3	4	5	6	7
ACC01	I can easily get the information to support my task from the program	1	2	3	4	5	6	7
ACU02	The are a number of errors and inaccurate information on the program	1	2	3	4	5	6	7
MEA02	The exact meaning of the information I found from the program is obvious	1	2	3	4	5	6	7
CUR01	I cannot get the information from the program that is current enough for my purpose	1	2	3	4	5	6	7
REL01	I can count on the program to be up and available when I need it	1	2	3	4	5	6	7

Positrol help system - Questionnaire for Operators

Table B-7 – Items chosen to measure perceived ease of use (EOU)

		No				Yes
EOU01	Is the program easy to use?	1	2	3	4	5
EOU02	Is learning from pictures and video easier than from text?	1	2	3	4	5
EOU03	Is the text instruction clear and easy to understand?	1	2	3	4	5

Table B-8 – Items chosen to measure perceived usefulness (U)

		No				Yes
U01	Do you think the program is useful?	1	2	3	4	5
U02	Do pictures and video help you do your job better?	1	2	3	4	5
U03	Will using the program help you do your job better?	1	2	3	4	5

Table B-9 – Items chosen to measure task-technology fit (TTF)

		No				Yes
TTF01	Do you feel that using the program will ensure you do the right task? (because content is up to date, accurate, etc)	1	2	3	4	5
TTF02	If you were a new operator, do you think the program is helpful to get you started?	1	2	3	4	5

Table B-10 – Items chosen to measure intention to use (IU)

		No				Yes
IU01	Will you use the program?	1	2	3	4	5
IU02	Will you recommend the program to your colleagues?	1	2	3	4	5

Questionnaire for Mfg Process Experts

Table B-11 – Items chosen to measure maintainability

		No				Yes
REU01	It is easy to create new content that is consistent in look and feel (guidelines available)	1	2	3	4	5
REU02	User interface components can be easily reused to create new content (template available)	1	2	3	4	5
CHA01	It is easy to update the content	1	2	3	4	5
CHA02	It is quick to update the links (handled by automatic hyperlink)	1	2	3	4	5
ANA01	It is easy to find faults (e.g. when a wrong picture is loaded)	1	2	3	4	5
ANA02	It is easy to find the broken links	1	2	3	4	5
STA01	Multimedia objects do not require regular updates of the drivers/players	1	2	3	4	5
STA02	The system is not affected by advances in computer technology (the use of mature technology like HTML)	1	2	3	4	5
TES01	Validating the system after modification is time consuming*	1	2	3	4	5
TES02	Changes in document formatting can be verified immediately (using style sheet)	1	2	3	4	5

* reverse-coded item

MM-MPS - Questionnaire for Mfg Process Experts

		No	Not sure	Yes
REU01	Do the authoring pages allow you to create new MPS that is consistent in look and feel?	1	2	3
REU02	Can you use the template provided to create multimedia objects for multimedia MPS?	1	2	3
CHA01	Do you need help from someone else to update the existing MPS?	1	2	3
CHA02	Do you need to embed hyperlink manually?	1	2	3
	If yes, is it easy?			
ANA01	Are you certain that both the content of MPS and the supporting multimedia objects have been approved by Config Control?	1	2	3
	If yes, how?			
ANA02	Can you easily find the broken links?	1	2	3
STA01	Do you need to use the latest plug-in, driver etc., to view multimedia objects?	1	2	3
STA02	Do you think the technology used in MM-MPS is mature enough?	1	2	3
TES01	Can you verify the changes that you've made immediately?	1	2	3
	If yes, how?			
TES02	Are you certain that changes in document formatting can be verified immediately?	1	2	3

Questionnaire used during Confirmatory Case Study

To what extent the following features are present (or adopted) in your methodology to develop your 'application'

Please circle the most appropriate number on the scale. '6' means the particular feature is present or has been adopted.

Features within development methodology	*Adopted*					
Formal methods to elicit user requirements and specifications for the application	1	2	3	4	5	6
Stakeholders' (users, process experts, developers) involvement during design process	1	2	3	4	5	6
The use of formal design techniques: in-house methodology, SSADM, SDLC, OO, E-R, etc.	1	2	3	4	5	6
Sole responsibility of the developer to generate, publish and maintain the content	1	2	3	4	5	6
Separation of content and presentation	1	2	3	4	5	6
Mechanism to allow other people to contribute the content easily	1	2	3	4	5	6
Documentation control and content approval of the application	1	2	3	4	5	6
Methods and strategies to maintain complex hyperlinks and to ensure the content remains up to date	1	2	3	4	5	6
Adoption of formal evaluation techniques to ensure usability of the application	1	2	3	4	5	6
Provision of other in-house success measures	1	2	3	4	5	6

In your organisation, what are the success factors for the development, implementation, deployment and maintenance of your 'application'?

1. Development
2. Implementation
3. Deployment
4. Maintenance

APPENDIX C

ManTIS and Motorola

Motorola was founded by Paul V. Galvin as the Galvin Manufacturing Corporation, in Chicago, Illinois, in 1928. Its first product was a 'battery eliminator' allowing consumers to operate radios directly from household current instead of the batteries supplied with early models. In the 1930s, the company successfully commercialised car radios under the brand name 'Motorola' a word suggesting sound in motion. During this period, the company also established home radio and police radio departments, instituted pioneering personnel programs and began national advertising. The name of the company was changed to Motorola, Inc. in 1947.

The decade of the 1940s also saw the company begin government work and open a research laboratory in Phoenix, Arizona, to explore solid-state electronics. By the time of Paul Galvin's death in 1959, Motorola was a leader in military, space and commercial communications, had built its first semiconductor facility and was a growing manufacturer of consumer electronics.

Under the leadership of Robert W. Galvin, Paul Galvin's son, Motorola expanded into international markets in the 1960s and began shifting its focus away from consumer electronics. The colour television receiver business was sold in the mid-1970s, allowing Motorola to concentrate its energies on high-technology markets in commercial, industrial and government fields. By the end of the 1980s, Motorola had become the premier worldwide supplier of cellular telephones and in 1996 produced the pocket-sized StarTAC cellular telephone.

Following the merger with General Instrument Corporation, Motorola became a leader in cable modems and set-top terminals. Today, Motorola is harnessing the power of wireless, broadband and the Internet to deliver embedded chip system level and end-to-end network communication solutions for the individual, work team, vehicle and home. Motorola also becomes the global leader in providing integrated communications solutions and embedded electronic solutions with the facilities in a number on countries.

The GSM (Global System for Mobile communication) Product Division facility in Swindon, UK, is dedicated to the research, development and manufacture of the GSM products, the standard developed in Europe and adopted world-wide for mobile telecommunication technology. The manufacturing facility in Swindon involves the production and assembly of mainly base transceiver stations (BTS) cabinets, simply called base stations hereafter. These base stations are used as the infrastructure to support cellular telecommunication providers all over the world. The demand for the product is such that the manufacturing process is continuous 24 hours a day, 7 days a week.

The manufacturing facility consists of a number of assembly lines starting from the front-end areas that mount the components into the printed circuit boards (PCB), to the systems testing area, as illustrated in Figure C-1.

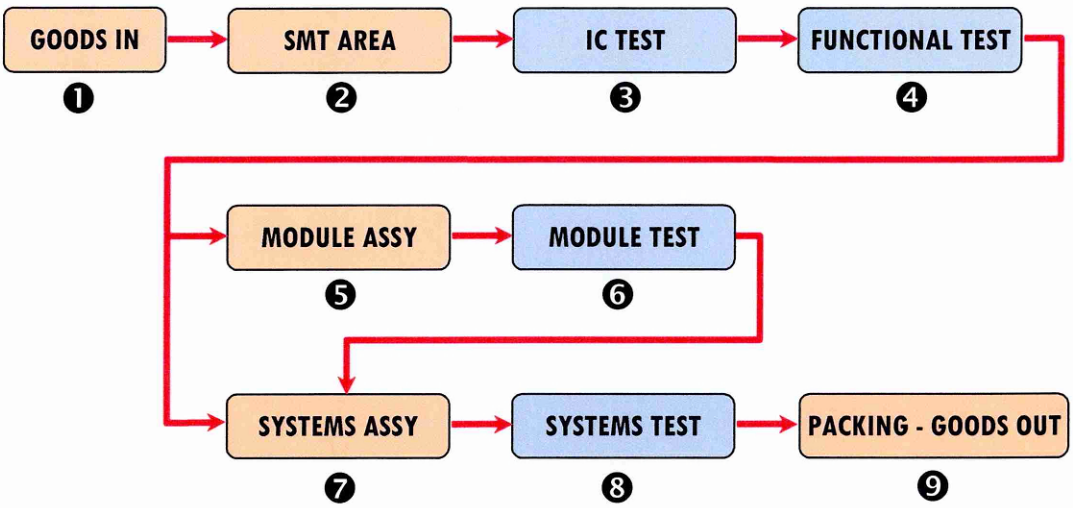


Figure C-1 – Simplified assembly line

1. **Goods In**. This area receives Printed Circuit Boards (PCB) arriving from vendor, ready for component population.

2. **Surface Mount Technology (SMT) Area**. PCB is inserted into the surface mount machines where components are pasted to the PCB by the competent placing machinery. The wave soldering machine then solders the components to the PCB.

3. **In Circuit (IC) Test Area**. The soldered PCB is tested on an IC test rig which will test the components on the populated PCB.

4. **Functional Test (FT) Area**. The PCB is tested on a functional test rig to test the functionality of the PCB.

5. **Module Assembly Area**. The PCBs are then assembled into modules in this area.

6. **Module Test Area**. The assembled modules are functionally tested. Some may contain a large number of circuit boards.

7. **Systems Assembly Area**: Modules and functionally tested circuit boards are assembled into so-called the system.

8. **Systems Test Area**: The assembled system is run through a functional systems test. In larger systems any faults that are found are debugged and repaired in situ (Real Time Debug and repair). Faulty modules and PCBs are sent back to modules and functional areas. On each testing area (indicated by number 3, 4, 6 and 8), there is an additional area where the faulty PCBs are debugged and repaired.

9. **Packing & Goods out**: Systems that pass the test are cleaned up and then packed into there packaging with manuals protective packing. Packaged systems are then distributed from the factory to depots and eventually to the customers.

Motorola has been continuously seeking innovative ways to improve efficiency of its testing processes and to increase productivity of the test personnel. One of the company's current initiatives is the application of web technology across its manufacturing facilities. Not only has this technology become so ubiquitous that attracts more users than any other IT applications, it has also demonstrated its potential in providing support to manufacturing operations. In order to assist the test personnel, including operators, technicians and engineers, to carry out a systems test of a base station, an industrial hypermedia task support system, called ManTIS (Manufacturing Test Information System), was developed as part of the research project sponsored by Motorola. ManTIS integrated various types of manufacturing and test information into a collection of structured information related to the task in-hand (Figure C-2). With this information integration, it was possible to:

- Restructure current manufacturing procedure documents so that task-related information would always be available when and where required, with substantial improvements in the test procedure.

- Supply the test personnel, including inexperienced operators, skilled technicians and engineers, with a comprehensive support system to carry out the test procedure and to retrieve information about various aspects of the base station testing process.

- Provide an interactive and comprehensive learning environment that could guide new operators to both carry out the test process from start to finish, with either minimal or no external training, and to learn other subjects of relevance.

- Support the company's strategic partnership programme with joint venture partners around the world, by providing online access to data sheets, quality documentations, training materials and other technical support documents.

ManTIS adopted the user-driven approach, allowing the test personnel to explore the system and to select only relevant information about a particular topic of interest. This encouraged novice users to navigate the system using their own methods, at their own

pace. Accordingly, a number of different access routes into the system, as well as the links between topics, were provided.

To accommodate the wide-range of the users' expertise, ManTIS was structured into modules (also known as *modes* within Motorola) that operate both online and offline. The on-line environment was used during the test process and acted as a support system, whereas the off-line environment could be used as a stand-alone training tool.

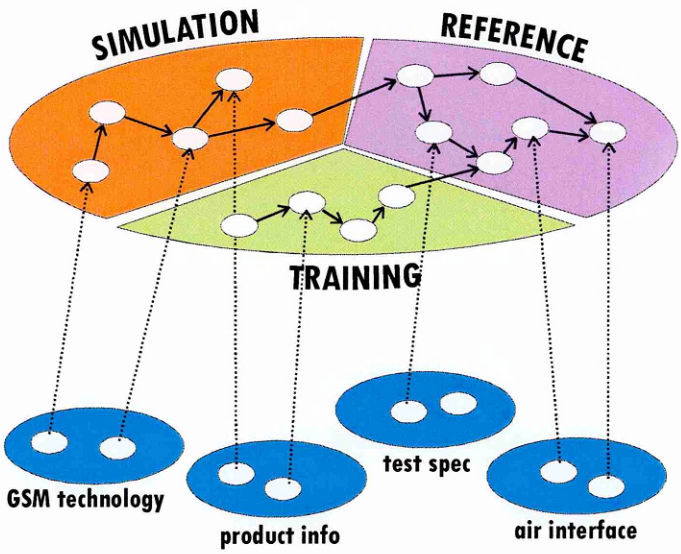


Figure C-2 – ManTIS concept

Table C-1 gives a comparison of the three modes of ManTIS.

Table C-1 – Comparison of ManTIS modes

Features	*Simulation mode*	*Reference mode*	*Training mode*
Technical information	Low	High	Intermediate
Method of access	Online and Offline	Online	Online
Navigation	Mostly sequential	Non-sequential	Hybrid
Interactivity	High	Low	Medium
Amount of text	Low	High	Medium
Multimedia objects	High	Low	Medium
Target user	Trainee operators	Technicians, engineers	Experienced operators

Background of Multimedia MPS

Manufacturing Process Specification (MPS) is Motorola's internal documentation to define manufacturing processes and thus to ensure consistency in the execution of manufacturing tasks. Every process is required to have an MPS approved and controlled by the documentation system. There are over 1,000 MPS documents, ranging from specifications on manufacturing the products to setting up work areas before commencement of work.

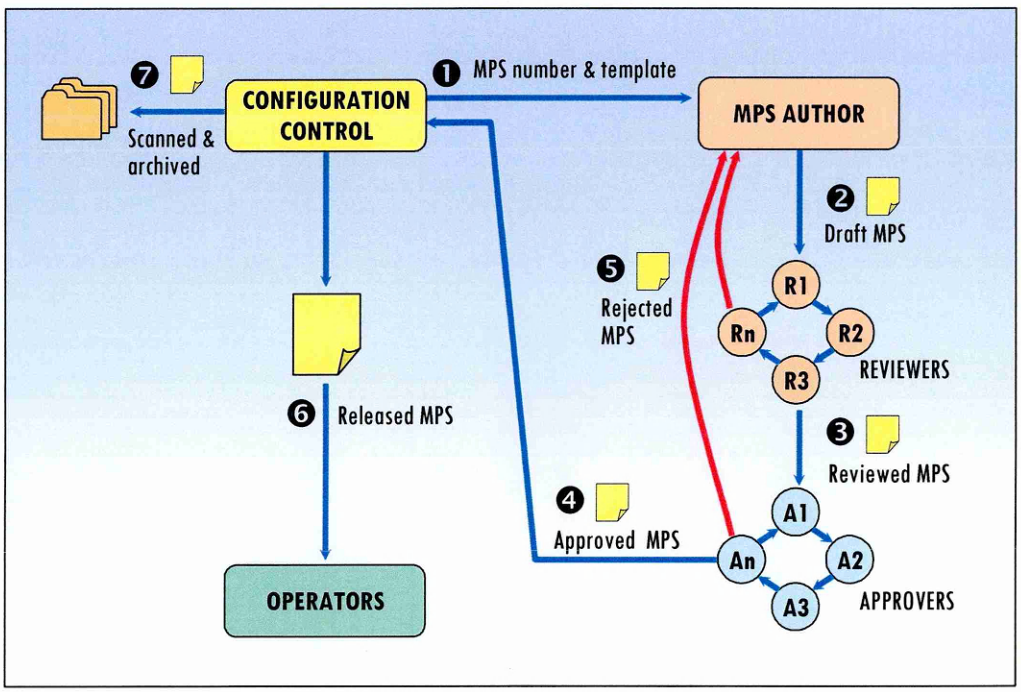


Figure C-3 – MPS authoring and approval workflow

Figure C-3 illustrates the paper-based MPS authoring and approval processes at Motorola. The MPS author writes an MPS using a predefined template created in Microsoft Word, issued by the Configuration Control department along with a unique MPS number. Once the author has finished writing an MPS, it is printed on paper and is physically circulated to a number of people for the review and approval process, also known as sign-off. The sign-off process consists of a number of sequential activities. First, the reviewers study the draft MPS. That MPS has to be approved by all reviewers before it is passed onto the approvers, otherwise the MPS will be returned to the MPS author. The MPS author needs to modify the MPS as required by the reviewers. Upon approval, all reviewers put their signatures on the MPS and pass the reviewed MPS to be distributed to the approvers.

Next, the MPS author circulates the reviewed MPS to a number of approvers, typically the managers or supervisors. Similar to the review process, the MPS will be returned to the MPS author unless all approvers accept it. Upon approval, the approvers also put their signatures on the MPS. This completes the approval processes, and the MPS has now been signed off. Finally, The MPS author hands over the signed-off MPS to Configuration Control to be released, by putting a stamp on the MPS. Following that, Configuration Control will also scan the approval sheet containing signatures from reviewers and approvers; archives the MPS; and finally releases the MPS to the shop floor.

The problems associated with the MPS and sign-off process can be described as follows:

- *Accessibility.* Since the MPS was in paper-based format, when someone needed an MPS, that person had to go to Configuration Control and ask for a paper copy.

- *Consistency.* The MPS template, as it existed, set out the logical sequence of information but enforced no rules for style. This meant that documents of widely varying appearance were issued, making the MPS less consistent.

- *Maintainability.* Updating paper-based MPS, with a large number of cross-references involved, was tedious and error prone.

- *Content currency.* As a result of poor maintainability, the content of paper-based MPS might not be up to date.

- *Comprehension.* With a large amount of text in paper-based MPS, shop floor operators tended to mark it with their own notes in order to make the instructions clearer. This could lead to discrediting the system and undermining management control.

- *Writing style.* Although the MPS was eventually scanned and converted into electronic format, the author wrote an MPS using the style for paper. The result was a lengthy document that was not suitable to be displayed electronically.

- *Document management.* Configuration control had to archive the MPS and to manage the scanned MPS. Although it was now an electronic document, no electronic document management was available to support the MPS delivery to the end user. The system was not intuitive and it was cumbersome to retrieve a particular MPS out of the system.

- *Issue control.* In this system, the MPS author needed to ask for the number from Configuration Control upon creation of new MPS. Although a unique MPS number needed to be issued manually by the Configuration Control Department,

this was a manual process, which possibly led to duplications and lack of control. Very often, people would acquire new numbers but not submit the MPS.

- *Electronic sign-off.* Assuming the electronic documentation was put in place, a way to electronically sign off the document had to be devised.

- *Traceability.* During the sign off process, paper-based MPS had to be circulated to the reviewers and approvers. Since there was no automatic workflow system, the author had to track down the current status of the MPS manually. Therefore, the duration of the sign off process could be considerably long.

In order to deal with the above problems, an electronic documentation system called *Multimedia MPS* (MM-MPS) is developed. The main purposes of the MM-MPS are to improve authoring process of the MPS; to increase management control of the documentation system; to speed up the approval process; to improve comprehension of current MPS and therefore to ensure the quality of task execution covered by the MPS.

Multimedia MPS incorporates multimedia objects such as pictures and digital videos, in addition to conventional text, to deliver the content of the MPS through the web (Figure C-4). These objects are stored on a database connected to the web server. When a user needs to retrieve an MPS from the system, that user will run a search on a particular MPS, for example MPS number 5012. The system uses this number as a key identifier to retrieve the associated multimedia objects and to populate these objects into a dynamically generated HTML page. This means no physical MPS is stored in the centralised repository. Instead, the multimedia objects are assembled into a full MPS document when required. This, in turn, will ensure the content currency.

The MM-MPS is also capable of generating automatic links to associated multimedia objects, which reduces the effort in updating both cross-references and content, thus increasing maintainability of the system.

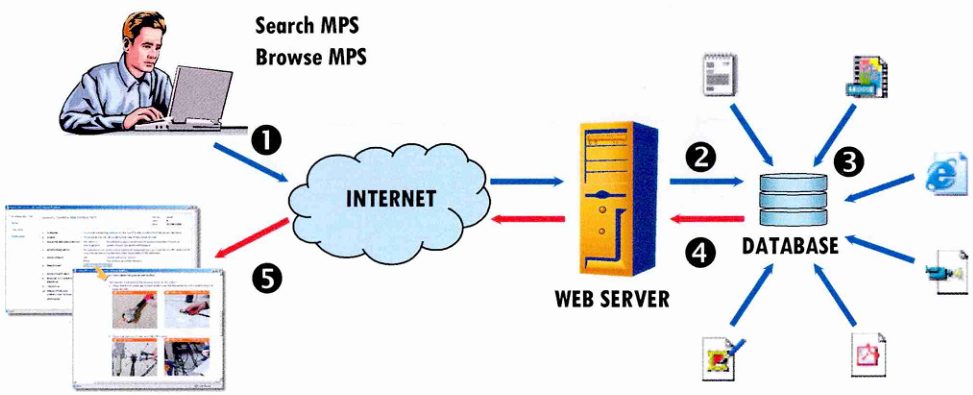


Figure C-4 – MM-MPS