Realising End-user Driven Web Application Development using Meta-design Paradigm

By Buddhima De Silva
Student Number: 15694195

A dissertation submitted in fulfillment of the requirements for the degree of Doctor of Philosophy.

December 2008
Dedication

This thesis is dedicated to my family and my friends for their continued support and encouragement. Without their encouragement, support, love and caring, it would not be possible to have completed this thesis.
Acknowledgement

First and foremost, I would like to thank my supervisor Professor Athula Ginige for his support during the course of my research. His ongoing commitment to expanding my research mindset and writing style was fantastic. Professor Ginige was always there with an encouraging word and healthy push to keep me moving forward when I was stuck no matter what the reason whether it was the enormity of the work, financial or personal problems. I would also like to thank my co-supervisor Dr. Robyn Lawson for her support for the case studies.

I would also like to extend a special thank you to my fellow researchers, especially Danny Liang and Makis Marmaridis, in AeIMS Research group. I’ve built up on their work and used the tools they implemented in my research. I wish to thank all the members of the research group for the help during surveys and user studies. Special thank to Mahesha Kapurubandara for referring me for the scholarship and all the support provided me during this study.

Grateful acknowledgement goes to the students in University of Western Sydney who were enthusiastic participants in the activities and helped to develop the tools. I would like to extend my gratitude to the members of Austool Ltd. and Innovative Technology Network (ITN) for their kind participation in the projects.

Special thanks to Vineeta Gunatilake and Tigger Wise who were very generous in proof reading the final draft.

Finally, I want to thank my family. I would never have aspired to research, and could certainly never have completed the PhD, without the encouragement and enthusiasm of my husband, Kushan. Special thanks need to go to my wonderful 3 year old daughter Losanie who left mum alone to finish her thesis. My parents Chandra and Wasantha deserve special mention for their support to look after
Losanie. Special thanks should go to my auntie, Vineeta and her family for taking care of my father in Sri Lanka while my mother came overseas to help me look after Losanie. Last but not least thanks should go to my parents in-law, Upali and Ranjanee, and brother in-law, Preshan. Without their willingly provided financial guarantee, I would not have been able to commence my PhD.
Statement of Authentication

The work presented in this thesis is, to the best of my knowledge and belief, original except as acknowledged in the text. I hereby declare that I have not submitted this material, either in full or in part, for a degree at this or any other institution.

........................................
(Signature)
Publications related to this thesis


Other Publications


# Table of Contents

Dedication........................................................................................................i

Acknowledgement..........................................................................................ii

Statement of Authentication............................................................................iv

Publications related to this Thesis.................................................................v

Other Publications.........................................................................................vi

Table of Contents...........................................................................................vii

List of Tables..................................................................................................xv

List of Figures..................................................................................................xvii

Abbreviations..................................................................................................xx

Abstract...........................................................................................................xxi

1 Introduction and Research Context.............................................................1

1.1 Introduction..............................................................................................1

1.2 Background ............................................................................................5

1.2.1 Development approaches......................................................................6

1.2.2 Development process for Web applications...........................................8

1.2.3 Component based eApplication Development and Deployment Shell

(CBEADS)........................................................................................................9

1.3 Motivation and Significance.....................................................................11

1.4 Research Problem and the Scope............................................................13

1.5 The Research Questions and Objectives................................................14

1.5.1 What are the different types of Web applications SME end-users require?

.....................................................................................................................14

1.5.2 What are the capabilities of end-users in relation to developing/ or

maintaining Web applications?.....................................................................15
1.5.3 How do the end-users conceptualise the Web applications? ................. 15
1.5.4 Based on the findings for the above three sub research questions, what are the requirements for a suitable end-user development approach for SMEs? ...... 16
1.5.5 How to create Web applications based on the identified requirements? . 16
1.6 Research Contributions............................................................................. 17
1.7 Research Approach.................................................................................... 18
1.7.1 Practical relevance .................................................................................. 21
1.7.2 Connection to prior theory ....................................................................... 21
1.7.3 Construction ............................................................................................ 21
1.7.4 Practical functioning of the solution......................................................... 22
1.7.5 Theoretical contribution .......................................................................... 23
1.7.6 Research strategies .................................................................................. 23
1.8 Thesis Overview .......................................................................................... 25
1.9 Conclusion .................................................................................................. 27

2 Related Work.................................................................................................... 28
2.1 Introduction .................................................................................................. 28
2.2 Key Terminologies ....................................................................................... 30
2.2.1 End-user .................................................................................................. 30
2.2.2 End-user development ............................................................................. 30
2.2.3 End-user developer .................................................................................. 32
2.2.4 Meta-design ............................................................................................ 32
2.2.5 Small to Medium Enterprises (SMEs) ..................................................... 34
2.2.6 Web application ....................................................................................... 35
2.2.7 Conceptual model ................................................................................... 36
2.2.8 Meta-model ............................................................................................ 36
2.3  End-user development ................................................................. 37
  2.3.1  End-user Development- managerial and social perspectives .......... 38
  2.3.2  End-user development concepts ............................................. 40
  2.3.3  End-user development activities ............................................. 43
  2.3.4  Conclusions and implications for realising meta-design paradigm .... 47
2.4  Psychology of Programming ......................................................... 47
  2.4.1  Mental model ............................................................................. 47
  2.4.2  Runtime programming ............................................................. 48
  2.4.3  Minimalism .................................................................................. 48
  2.4.4  Conclusions and implications for realising a meta-design paradigm .... 49
2.5  EUD Web ....................................................................................... 50
  2.5.1  Different types of Web applications required for SME end-users to
         support their businesses ................................................................. 50
  2.5.2  EUD Web Tools ......................................................................... 54
  2.5.3  Conclusions and Implications for realising meta-design paradigm .... 57
2.6  Web Engineering ............................................................................ 60
  2.6.1  Web development tools ............................................................. 60
  2.6.2  Web development frameworks .................................................. 63
  2.6.3  Model driven approaches .......................................................... 64
  2.6.4  Meta-model of Web applications ............................................... 66
  2.6.5  Conclusions and implications for EUD Web development methodology 67
2.7  Conclusion ....................................................................................... 67
3  Identifying the Requirements to Facilitate End-user Development .. 69
  3.1  Introduction .................................................................................... 69
  3.2  Grounded Theory ........................................................................... 71
4.2.2 Navigation model ................................................................. 111
4.2.3 Composition model ............................................................. 112
4.2.4 Access Control model .......................................................... 114
4.2.5 Personalisation ................................................................. 114
4.3 Aspects required to model data intensive Web applications ..... 115
4.3.1 Data model ........................................................................ 117
4.3.2 Navigation model ................................................................. 118
4.3.3 Composition Model ............................................................. 119
4.3.4 Action model .................................................................... 120
4.3.5 Access control model .......................................................... 122
4.4 Aspects required to model workflow intensive Web applications ... 123
4.4.1 Workflow model ................................................................. 123
Figure 4.18: Workflow model for Notification of Intention .......... 125
4.5 Study on end-user specifications ............................................. 125
4.6 Conclusion ........................................................................... 127

5 Meta-design Paradigm and Meta-model ............................. 129
5.1 Introduction ........................................................................ 129
5.2 Meta-Model ....................................................................... 132
5.2.1 Meta-model properties ..................................................... 134
5.3 Meta-model of Process-centric Web applications .................... 135
5.3.1 Mapping of aspects from conceptual model to meta-model .... 136
5.3.2 Shell level ....................................................................... 138
5.3.3 Application level ................................................................. 141
5.3.4 Function level ................................................................. 142
<table>
<thead>
<tr>
<th>Content Management System</th>
<th>234</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workflow Management System</td>
<td>239</td>
</tr>
<tr>
<td>Virtual Team based Access Control</td>
<td>242</td>
</tr>
<tr>
<td>SME Questionnaire 1</td>
<td>243</td>
</tr>
<tr>
<td>SME EUD Workshop notes</td>
<td>245</td>
</tr>
<tr>
<td>SME EUD Questionnaire 2</td>
<td>247</td>
</tr>
<tr>
<td>End-user Developer Interview Questionnaire</td>
<td>248</td>
</tr>
<tr>
<td>EUD Questionnaire 1</td>
<td>251</td>
</tr>
<tr>
<td>EUD Assignment 1</td>
<td>258</td>
</tr>
<tr>
<td>EUD Questionnaire 2</td>
<td>262</td>
</tr>
<tr>
<td>End-user Specification Questionnaire 1</td>
<td>264</td>
</tr>
<tr>
<td>End-user Specification Questionnaire 2</td>
<td>270</td>
</tr>
<tr>
<td>Assignment 1- Feedback Form</td>
<td>274</td>
</tr>
<tr>
<td>Tutorial -Week 9</td>
<td>278</td>
</tr>
</tbody>
</table>
List of Tables

Table 1.1: Original Contributions from the Research ................................................................. 17
Table 1.2: Research Strategies .................................................................................................... 23
Table 1.3: Summary of Chapters ............................................................................................... 25
Table 2.1: Definition of Terms: SMEs and MEs (adapted from European Union, 2003
(Gamage, 2003) .......................................................................................................................... 34
Table 2.2: External and Internal processes in Roadmap adopted from Kazanis (2004) .......... 52
Table 2.3: Definitions for different types of Web applications .................................................. 53
Table 2.4: Comparison of EUD Web Tools .............................................................................. 59
Table 2.5: Tools groups supporting Web application development ......................................... 60
Table 3.1: The assignment tasks to create/ maintain Web site .................................................. 75
Table 3.2: Website development and Maintenance Tasks .......................................................... 77
Table 3.3: Task Completion rates ............................................................................................. 83
Table 3.4: Difficulty of tasks and rating description ................................................................ 83
Table 3.5: End-user development Activities in OCAS .............................................................. 86
Table 4.1: Summary of Web application modelling approaches .............................................. 104
Table 4.2: Site level Presentation properties .......................................................................... 111
Table 4.3: Composition Model of Actif site ........................................................................... 113
Table 4.4: Usage of aspects in end-user specifications ............................................................. 126
Table 4.5: Summary of Modifications to WebML .................................................................... 127
Table 4.6: Models and Attributes of different aspects .............................................................. 128
Table 5.1: Functions to manage Process Centric Applications ............................................... 147
Table 5.2: Functions to manage Information-centric Web applications .................................. 154
Table 5.3: Meta-model activities at different complexity levels ................................................. 157
Table 6.1: Factors present with the tool makers ...................................................................... 170
Table 6.2: The guidelines to meta-design approach.................................................................171

Table 6.3: Maintenance Activities ..........................................................................................173

Table 6.4: SMEs feedback on routine level development activities relate to aspects.............174

Table 6.5: Categorisation of different Implementation Domain Concepts.............................177

Table 6.6: Activities required in developing Web applications using meta-model ...............182

Table 6.7: Background Study-Students responses to the conceptual model and aspects......187

Table 6.8: Week by week unit schedule..................................................................................187

Table 6.9: Questionnaire 2-Students responses to the conceptual model and aspects of Workflow intensive Web applications.................................................................189

Table 6.10: Assignment 1- Usage of the aspects for the development of the Web sites.......191

Table 6.11: Strategies to be used in teaching modelling........................................................193

Table 6.12: Strategies for End-user development..................................................................193

Table 7.1: Aspects required to specify different types of Web applications.........................199
List of Figures

Figure 1.1: Possible Web development Approaches (J. A. Ginige, De Silva, & Ginige, 2005) ..............................................................................................................................................7
Figure 1.2: Overall CBEADS Architecture (A. Ginige, 2002) .............................................................................................................10
Figure 1.3: Elements of Constructive Research (Lukka, 2003, p. 246) .........................................................................................19
Figure 1.4: Overall research process ...........................................................................................................................................20
Figure 2.1: Related Areas adapted from Rode (2005) ......................................................................................................................28
Figure 2.2: Managerial and Social Perspectives (Adapted from (Fischer, Ye, Sutcliffe, & Mehandjiev, 2004)) ....................................................................................................................39
Figure 2.3: The eTransformation Roadmap adopted from Kazanis (2004) ..................................................................................51
Figure 3.1: WYSIWYG Editor ......................................................................................................................................................78
Figure 3.2: User Interface-Create Web site ..................................................................................................................................79
Figure 3.3: Perception of benefits and motivations for EUD ........................................................................................................80
Figure 3.4: Barriers to EUD ......................................................................................................................................................81
Figure 3.5: Experience in Computer Software applications ........................................................................................................81
Figure 3.6: Experience in Web application development .............................................................................................................82
Figure 4.1: Web Modelling approaches (Schwinger & Koch, 2006) .................................................................................................97
Figure 4.2: WebML Content units (Brambilla, Comai, Fratenali, & Matera, 2008) ........................................................................105
Figure 4.3: WebML operation units (Brambilla, Comai, Fratenali, & Matera, 2008) .................................................................106
Figure 4.4: BPMN main construct (Marco, Stefano, Piero, & Ioana, 2006) .....................................................................................106
Figure 4.5: An example Site View which gives the access control aspect (Brambilla, Ceri, Fratenali, & Manolescu, 2007)..................................................................................................................108
Figure 4.6: Actif Toolcraft.com Home page....................................................................................................................................110
Figure 4.7: Site Level Template of Actif ........................................................................................................................................113
Figure 4.8: Primary navigation of Actif site .......................................................................................................................................112
Figure 4.9: Access Control model..................................................................................................114
Figure 4.10: A product page from Bosco site ..............................................................................116
Figure 4.11: A sub product page from Bosco site.......................................................................116
Figure 4.12: Structure Schema used in Bosco Catalogue............................................................117
Figure 4.13: Data model ...................................................................................................................118
Figure 4.14: Navigation Model of Product Catalogue.................................................................119
Figure 4.15: Composition model of the sub product page ........................................................120
Figure 4.16: Action model of product catalogue.........................................................................121
Figure 4.17: Access Control Model of Product Catalogue.........................................................122
Figure 4.18: Workflow model for Notification of Intention .....................................................125
Figure 5.1: Three level Meta-model...............................................................................................133
Figure 5.2: Mapping of process centric Web applications from conceptual level to the meta-model........................................................................................................................136
Figure 5.3: Meta-model for process-centric Web applications....................................................138
Figure 5.4: Shell level of the meta-model......................................................................................139
Figure 5.5: Application Level of the meta-model........................................................................141
Figure 5.6: Function level of the meta-model...............................................................................143
Figure 5.7: Mapping the Aspects of a process centric Web application to the implementation........................................................................................................146
Figure 5.8: Mapping the aspects from Conceptual model to the meta-model........................149
Figure 5.9: Meta-model for Information centric Web applications...........................................150
Figure 5.10: Shell level in meta-model for Information centric Web applications.................151
Figure 5.11: Application level in meta-model for Information centric Web applications.....151
Figure 5.12: Mapping the Aspects of a information centric Web application to the implementation........................................................................................................153
Figure 6.1: Tool makers position on the eTransformation roadmap (Arunatileka, Ginige, Hol, & Lawson, 2005) ............................................................................................................169

Figure 6.2: knowledge in different technologies used in Web application development ......178

Figure 6.3: Mean values for Capability of using implementation domain concepts.........179

Figure 6.4: Easiness of creating with and with out meta-model ...........................................183

Figure 6.5: Action Research Cycle .................................................................................................185

Figure 7.1: Research Summary .......................................................................................................197
### Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AeIMS</td>
<td>Advanced enterprise and Information Management Systems</td>
</tr>
<tr>
<td>BPMN</td>
<td>Business Process Modelling Notations</td>
</tr>
<tr>
<td>DIY</td>
<td>Do It Yourself</td>
</tr>
<tr>
<td>EUD</td>
<td>End user Development</td>
</tr>
<tr>
<td>FTP</td>
<td>File Transfer Protocol</td>
</tr>
<tr>
<td>SME</td>
<td>Small to Medium Enterprises</td>
</tr>
<tr>
<td>ICT</td>
<td>Information and Communication Technologies</td>
</tr>
<tr>
<td>IT</td>
<td>Information Technologies</td>
</tr>
<tr>
<td>OCAS</td>
<td>Online Course Approval System</td>
</tr>
<tr>
<td>OMT</td>
<td>Object Modelling Technology</td>
</tr>
<tr>
<td>UI</td>
<td>User Interface</td>
</tr>
<tr>
<td>UML</td>
<td>Unified Modelling Language</td>
</tr>
<tr>
<td>UWS</td>
<td>University of Western Sydney</td>
</tr>
<tr>
<td>WfMC</td>
<td>Workflow Management Coalition</td>
</tr>
<tr>
<td>WYSIWYG</td>
<td>What You See Is What You Get</td>
</tr>
<tr>
<td>ISP</td>
<td>Internet Service Provider</td>
</tr>
</tbody>
</table>
Abstract

Small to Medium Enterprises (SMEs) need to use Information and Communication Technologies (ICT) to enhance their business processes to become competitive in the global economy. When an information system is introduced to an organisation it changes the original business environment thus changing the original requirements. This in turn will change the processes that are supported by the information system. Also when users get familiar with the system they ask for more functionality. This gives rise to a cycle of changes known as co-evolution. In addition, SMEs have budget constraints which make the problem associated with co-evolution worse. One solution to overcome this situation is to empower end-users to develop and maintain the information systems.

Within the above context the work presented addresses the following research question:

“How to support SME end-users to develop and / or maintain Web applications to support their business processes?”

There are two main components to this question: What are the requirements of a suitable end-user development approach for SMEs and how to create the Web applications based on the requirements. The requirements of a suitable end-user development approach can be established by identifying the different types of Web applications required by SMEs, the capabilities of end-users in relation to developing and / or maintaining Web applications and how they conceptualise the Web applications.
The literature review is conducted to discover different types of Web applications required by SMEs and to identify a suitable end-user development approach and tools that can support the development of these various types of Web applications. According to the literature survey, the main types of Web applications required by SMEs can be categorised as information centric Web applications (Simple Web sites which focus on effective presentation of unstructured information), data intensive Web applications (the focus is on efficient presentation and management of structured data such as product catalogue) and workflow intensive Web applications (The focus is on efficient automation of business processes such as an order processing system). The literature on end-user development shows that the existing end-user development approaches are focused on specific types of Web applications. The frameworks and tools in the Web development discipline mainly target experienced Web developers. Therefore a gap is identified as “there are limited end-user development approaches for developments of different types of Web applications which are required by SMEs to IT enable their businesses”.

The capabilities of SMEs in relation to Web application development were identified based on a study conducted with a group of SMEs. This study first surveyed the SMEs experience and knowledge in relation to Web application development and their attitude towards end-user development. Then their capabilities relating to Web application development were studied in a hands-on session to develop a Web site. The second study is conducted with administrative staff members involved in development of a Web application. This study helps to establish the requirements of a suitable end-user development approach from the
point of view of the end-user developers. This study on end-user development observed the different activities carried out by end-users. Then the end-user was interviewed to identify the issues and benefits of end-user development in the project. Following that, a set of requirements for the end-user development approach was derived based on the findings from these two studies and the related literature: 1) A need to support different types of Web applications required by SMEs; 2) A need to support the specification of Web applications at the conceptual level; 3) A need for a common data repository to store the data used in different applications within the organisation; 4) Providing a common login to all applications within the organisation; 5) Stripping a balance between Do it Yourself (DIY) and a professional developer that allows end-users to do the activities they are capable of while getting help from a professional developer to do the difficult tasks.

The conceptual aspects of different types of Web applications (information centric, data intensive and process intensive) were identified based on a literature survey of existing conceptual modelling approaches. This set of aspects was refined by modelling selected Web applications for each type of Web application. The aspects needed to specify different types of Web applications are: presentation, data, task, workflow, access control, navigation, and personalisation. Then the usage of these aspects in a set of end-user specifications was analysed. This study reveals that end-users only focus on some of these aspects such as data and process to specify the applications. Therefore, another requirement for the development approach was identified – a need to support development of Web applications with minimum aspects.
A meta-design paradigm based on the meta-model of Web applications is proposed to support the identified requirements. A meta-model of Web applications is developed based on the patterns of different types of Web applications. A component based Web application development framework called CBEADS (Component based eApplication Development and Deployment Shell) was extended to support the meta-model based development approach. Web applications can be created by populating the values for the attributes of the meta-model which are related to the attributes of different aspects of the Web applications at the conceptual level. The meta-model is organised into three levels: shell level, application level, and function level. Aspects common to many web applications are modelled at the shell level. The data model and user model are stored and managed at the shell level. This supports the requirements of common data repository and the common login to all applications. The aspects common to a web application are modelled at the application level. The function specific aspects required to implement the functionality of the Web application are modelled at the function level. The meta-model has two properties called overriding and inheritance. Inheritance property allows developing Web applications with minimum aspects. The activities required to develop the Web applications in a framework supporting the meta-model are grouped into three levels based on the complexity of these tasks named routine level, logical level and programming level. These different levels together with the overriding property help to balance between DIY and a professional developer.

The meta-design paradigm is practically evaluated with a group of users including SMEs and students. The studies establish strategies for the success of the meta-design paradigm such as characteristics of individuals, facilitation and infrastructure.
The original contributions of this thesis enhance the field of end-user development by providing a new end-user development approach that can be used by business end-users to develop web applications. More importantly the major contributions of this research provide a practical approach that can be used particularly by SME end-users with little or no previous experience in web application development. Significant research contributions are made in the following four areas:

1) Establishing requirements for an end-user development approach suitable for business users.

2) Identifying a set of aspects required to model different types of Web applications at the conceptual level.

3) Developing a meta-design paradigm based on the meta-model of different types of Web applications

4) Developing the strategies for successful use of the meta-design paradigm.

Key words: end-user development, meta-design, meta-model, web applications, evolution, conceptual modelling
1 Introduction and Research

Context

1.1 Introduction

With the objective of sharing documents between researchers, Tim Berners-Lee designed the Web as a collaborative tool (Berners-Lee, 1996). Original information centric Web sites have evolved into database centric Web applications and gone beyond, to create Web applications to support business processes, collaborative work and social networking. With these developments, Information and Communication Technologies have become critical to support businesses; not only to carry on business activities, but also to make their presence felt in the world, through the Web. Ubiquity and ease-of-access to the Web have contributed towards making Web application the first choice of technology to disseminate information and automate business processes.

In traditional approaches to develop Web applications, an organisation that requires a new information system, first analyses its requirements. If it wishes to go for a custom-made information system the design specification is produced and the system gets developed, tested and deployed. Or else an organization buys a product to match their requirements and adapts their processes accordingly. The system then is maintained until decommissioned. However, the author observes that many such systems that were deployed within the University and in client organisations were short-lived and after a
period of time no longer met user requirements. On further investigation, it revealed three reasons for this failure. They are as follows:

1) Changes to the original business environment resulting from the introduction of the information system.

2) Changes to the processes that the information system supports.

3) Constant user requests for more functionality as they get used to the system.

It is necessary to modify the existing system to meet evolving requirements since it is impossible to know what will be required in advance (Nardi, 1993). Again the system needs to be changed responding to the new requirements that emerged due to changes in the way users use the system or changes in current environment or the technology. This gives rise to a cycle of changes known as co-evolution (M. Costabile, F., Fogli, Marcante, & Piccinno, 2006). Meta-design is proposed as a solution to co-evolution (M. Costabile, F., Fogli, Marcante, & Piccinno, 2006; M. Costabile, F., Fogli, Mussio, & Piccinno, 2005a). A fundamental objective of a meta-design paradigm is to create socio-technical environments that empower users to engage actively in the continuous development of systems rather than be restricted to the use of existing systems. A Meta-design paradigm consists of objectives, techniques, and processes for creating new media and environments allowing ‘owners of problems’ (that is, end-users) to act as designers (Fischer & Giaccardi, 2004; Fischer & Scharff, 2000; Fischer, Ye, Sutcliffe, & Mehandjiev, 2004). Then the end-users become the end-user developers. End-user developers are defined as people who develop applications without formal background in Information Technology to support their main job functions (Myers, Ko, & Burnett, 2006).
End-user development has become popular with the introduction of easy to use development tools and metaphors. Rittenberg, Senn & Bariff (1990) have observed end-users developing applications to support a wide range of decision making activities and business processing using excel. Researchers have reported several case studies on end-users using Web application development tools such as frontpage (Nelson & Todd, 1999; O’Brien, 2002; Rosson, Ballin, & Heather, 2004). Sophisticated development tools and processes are required to develop web applications supporting business processes to go beyond simple web sites. Researchers and commercial web development tool vendors are coming up with specific technologies and tools such as WYSIWYG editors, Wizards, programming by example, and programming by demonstration to help end-users (Lieberman, 2001; Rode, 2005; Wolber, Su, & Chiang, 2002).

Thus far the primary objective of end-user development tools is to simplify the construction process (e.g., page creation and linking, database access), not focusing on analysis or design (Rosson, Sinha, Bhattacharya, & Zhao, 2007). The fundamental challenges to end-user development have gradually shifted from basic syntactic adjustment required to help during construction towards semantic challenges including the need to convey an understanding of design and engineering principles relevant to end-users (Repenning, 2007). A more ‘natural’ development process should be devised to support not only during construction, but for the whole life cycle of the application development for the success of end-user development. Natural development implies that people should be able to work through familiar and immediately understandable representations that allow them to easily express and manipulate relevant concepts, and thereby create or modify software artefacts (Paterno, 2003).
This thesis presents a meta-design paradigm based on a meta-model with conceptual level modelling primitives of web applications as the inputs for a more natural approach to end-user driven web application development. The inputs to the meta-model are derived from the conceptual models of different types of Web applications. By encapsulating low-level technical aspects, such a development framework will enable end-user development (Beringer, 2004).

This thesis presents the conceptual aspects of different types of Web applications which was used as input into the meta-model. These aspects were identified in a literature survey on existing conceptual modelling approaches. Then, this set of aspects was refined by modelling selected Web applications for each type of Web application. The author analysed a set of proposals submitted by users in the university to develop or modify existing information systems. The analysis of end-user specifications revealed that end-users think of the systems as ways to store data and then manipulate data. They are not concerned about critical aspects such as presentation and hypertext at the application domain, likely to surface as usability issues of the system when in use. However, when they become the owners of the system they need to create or modify the system. Then they also need to consider other aspects such as the hypertext model. Therefore, the proposed end-user development approach should generate applications with the specification of minimum aspects by providing default values for some of the aspects.

This thesis demonstrates the mapping of conceptual model aspects into the meta-model which leads to a more natural development approach. The meta-model is organised into three levels: shell level, application level, and function level. Activities to create the Web applications based on the meta-model are grouped into three different complexity levels: routine level, logic level and programming level to tackle the problem of wide ranging
skill levels among the users, or the affordability of learning complex tasks as suggested by
different researchers such as Beringer (2004) and Burton et al. (1984). This enables
viewing the software as a medium to capture knowledge rather than an end-point as
suggested by researchers (Armour, 2000; A. Ginige, 2002; Shneiderman, 2003).

The objective of the meta-model is to enable end-users to develop/ maintain the
different types of Web applications by specifying the aspects of the application at the
conceptual level. The meta-design paradigm was evaluated using a series of qualitative
studies which involved SMEs and students. These evaluations led to the establishment of
strategies for end-user development tools and the conclusion that the meta-design
paradigm can be used to help SMEs to develop and maintain their Web applications.

This chapter presents the basis for the thesis. First the background to the research into
end-user development is discussed. The significance of further research into end-user
development of web applications in the context of SMEs is established, further leading
on to an explanation of the research aims, objectives, research questions and the research
methodology. This is followed by the contributions of the study. Finally, the outline of
the thesis is presented.

1.2 Background

Western Sydney is the fastest growing economic region in Australia, and consists of 10% of
the population. From about 72,000 businesses in the region, over 80% are Small to
Medium Enterprises (SMEs), with manufacturing among the largest industry sectors. The
percentage of manufacturing SMEs with a website in the region was 65% in 2003
(Khandelwal et al, 2004). Major barriers to adopt ICT in their businesses are lack of IT
expertise, cost and time to implement (Lawson, Arunatileka, A. Ginige & Hol, 2005).
Advanced Enterprise and Information Management Systems (AeIMS) research group at University of Western Sydney is working with SMEs in the region to use Information and Communication Technologies (ICT) to enhance their business processes to become competitive in a global economy (A. Ginige, 2004, 2006). In this research they have come up with a methodology for successfully transforming SMEs to eBusinesses called cTransformation roadmap (A. Ginige et al, 2001) and a strategic model to undertake the transformation called 7E Model (Arunatileka & A. Ginige, 2003). In this research another challenge the researchers faced was to find a way to develop Web based business applications that can evolve with changing business needs (A. Ginige, 2001). Also such applications need to develop rapidly as well as in a cost effective manner (A. Ginige, 2002). The development approach also needs to reduce the gap between what the users actually want and what is being implemented in terms of functionality (Epner, 2000).

Empowering end-users to develop Web applications is one solution to the issues mentioned above. If end-users have the ownership of the application, they are in a position to change their application once the requirements change. Also it will be cost effective and reduce the development time if end-users are familiar with their requirements as the owner of the problem. The next sections describe the different extremes in development approaches and where the meta-model based meta-design paradigm is going to be placed. This is followed by a brief explanation on the development process for Web applications. The last section introduces the Component based eApplication Development and Deployment Shell (CBEADS) which gets extended to support the meta-design paradigm.
1.2.1 Development approaches

A web application can be developed in one of the three approaches shown in Figure 1.1 (J. A. Ginige, De Silva, & A. Ginige, 2005). In the traditional approach (see (a) in Figure 1.1) an analyst is engaged to specify the application requirements, designers then model the system and developers implement it. In reality all of the above tasks are done either by an individual or a group of people, depending on the complexity of the system. The other extreme (see (c) in Figure 1.1) is to use a tool that allows users to attempt to do the task by themselves. WYSIWYG type Web development tools such as FrontPage and Dreamweaver make it possible for end-users to develop simple Web applications without doing any coding or relying on a developer.

![Figure 1.1: Possible Web development Approaches (J. A. Ginige, De Silva, & A. Ginige, 2005)](image-url)
Both of the above approaches (a) and (c) have their own problems. The success of resolving the Web application development issues in the first method depends on many factors such as how well the users communicate requirements to the analyst, the skills of the design and development team, the appropriateness of methodologies used and the flexibility of the design to incorporate future needs. Also this approach is expensive when the system evolves. On the other hand, option (c) makes it possible for users to develop these applications themselves. Hence, it can be assumed that the issues of communicating user requirements to analysts are eliminated. However, most tools that are available for end-user Web application development are either not capable of developing different types of Web applications as required or are way too complex to be used by end-users as discussed in chapter 2.

Therefore, the hybrid approach (b) is suggested for the proposed meta-design paradigm. In his approach developers create Components, which end-users can customise to create and deploy Web applications. This hybrid approach also supports incremental development. End-users can develop applications when required and also maintain them. Developers can keep adding new components when existing Components cannot meet end-user needs.

### 1.2.2 Development process for Web applications

A, Ginige. (2008) has divided the development process of a Web application into two domains: 1) Application domain; and 2) Implementation domain. An application domain consists of the concepts derived from the requirements of the web application. The concepts in the application domain such as data and processes form the conceptual model which solves the problem which the requirements present. Johnson and Henderson (2002) defines the conceptual model as a high-level description of how a system is organized and operates. The conceptual model helps to check whether the proposed
solution meets the requirements of the application as originally conceptualised. Once the conceptual model is established and verified, then a logical architecture model should be developed using the concepts of the implementation domain. In the component based development approach these concepts could be an access control module, view generator, workflow engine, object or data builder which can be implemented using the components. The specific concepts related to the architecture in the implementation domain can be called the background domain as specified in modelling spaces (Unhelkar, 2003). The background domain consists of the basis for the architecture and the company’s infrastructure, thereby providing constraints as well as facilitating opportunities for the solution to be modelled appropriately. For Web-based applications the background domain is assumed to be the Web infrastructure with client server architecture.

### 1.2.3 Component based eApplication Development and Deployment Shell (CBEADS)

The high level architecture of CBEADS (A. Ginige, 2001, 2002; A. Ginige, Liang, Marmaridis, A. Ginige, & De Silva, 2007) is shown in Figure 1.2. The shell itself is made of components. These components can be grouped into two major sub systems. The first sub system is Core CBEADS that provides the overall framework within which different business applications can be developed and deployed. It consists of a security module, system components, system database and a workflow component. The second sub system consists of various applications deployed within the shell. The detailed application architecture was created based on the Model-View Controller (MVC) architecture pattern (Reenskaug, 1979). The MVC architecture pattern separates the concerns in an application using three elements - model, view and controller. The model defines the internal data structures of the application, the view defines how the model is
rendered to the user, and the controller performs the actual actions in the application that affect the model (Lloyd, Rimov, Hamel, & al., 1999).

![Diagram of CBEADS Architecture]

Figure 1.2: Overall CBEADS Architecture (A. Ginige, 2002)

In CBEADS, all user interactions coming from Web browsers on users’ computers are passed through a security module that carries out the authentication as well as checks what applications and what functions within these applications the user is permitted to access. First, a user needs to login to the system. After verifying the user id and the password the user is given a personalised home page. This page displays the applications and the functions within these applications that the user is permitted to access. The user can access these functions by clicking on the links. As an added security measure before the requested function is activated the security module again checks whether the user is permitted to use the requested function.

User and Access Control Management is one of the system applications that supports the applications built on the framework. This application has functions to create user groups,
allocate different functions within other applications to user groups, create users and allocate them to user groups. This user and access control management application can be considered as a configurable component to suit the needs of the business applications. The user and function access data are stored in the system database.

There is a special function among CBEADS system components that can be used to create new functions. This ability to create new functions enables CBEADS to grow. These functions that get created can be grouped into forming new applications.

A set of tools to facilitate development of applications uses the function to create new functions. These tools range from simple code editors and tools to create databases to ‘smart tools’ for generating Smart Business Objects (SBO) (Liang & A. Ginige, 2006), User Interface Generator (UI Renderar) (Liang & A. Ginige, 2007), Workflow Engines (WFE) (Liang, Marmaridis & A. Ginige, 2007) and State Dependent Access Control (VTMAC) (Marmaridis & A. Ginige, 2007). The ‘smart tools’ embed some of the computer domain specific knowledge into the tool. These tools facilitate the creation of the applications within the CBEADS framework.

1.3 Motivation and Significance

Why is “end-user Web application development” desirable and important in SME context? Empowering end-users to develop Web applications will have several important consequences. These are described below.

Most importantly, end-users will no longer depend solely on developers to create custom Web applications, enabling the production of a wide range of applications, which can evolve to match the end-users’ requirements. This will reduce the development cost and
give the SMEs control over the applications they use. A survey was carried out by the author with twenty four SMEs in the Western Sydney region in Australia to identify the applicability and advantages of end-user development for Small to Medium Enterprises (SMEs). The benefits of the ‘Do it yourself’ approach include the reduction of cost and getting the application to match their requirements (Hason, 2001). The same study further revealed that 80% of SMEs were willing to participate in end-user development. The details of this study is described in chapter 3.

Secondly, the work processes may become more efficient as individual personnel are able to develop or change the application to suit their job function. End-user development helps to achieve higher productivity of labour (Paterno, Klann, & Wulf, 2003). During the development of a complex Web application for University of Western Sydney called Online Course Approval System (OCAS) (J. A. Ginige & A. Ginige, 2007), it was identified that the end-users wanted the interfaces with labels and help information to match their every day terminology. This would reduce their time in getting used to the terminology used in the new system. If a system adapts according to the users requirements, there are likely to be only a few changes to the processes when a new system is introduced. Also the system can be used with minimum training. If the users feel they have ownership of a system, they tend to use the system more effectively.

Thirdly, the increased number and diversity of people creating Web applications will promote innovation, as Deshpande and Hansen suggest: “[releasing] the creative power of people.” (Deshpande & Hansen, 2001). The users always find new ways of doing their jobs and using the Web applications. For example, AeIMS research group has developed a Web site for a SME running a plant nursery to get Web presence. This person has included his Web address in the label, which enables his customers to directly contact
him after purchasing the plants and which is seen as a likely move to improve his business. If the business owners become designers as envisioned in the meta-design paradigm rather than being restricted to be just stake holders, they will be in a better position to release their creative power and to get competitive advantage.

Finally, in addition to empowering end-users to pursue new goals, it will help novice developers create Web applications that are more secure, cross-platform-compatible, and universally accessible (Rode, 2005). It helps users to develop and deliver high quality Web applications without much training and experience.

1.4 Research Problem and the Scope

Ginige, A. (2002) suggests that computers should be viewed as a way of capturing knowledge instead of as a means to create specific end products in order to meet the needs of business users in relation to Web applications to support their business processes. Fischer, G. also supports this view of end-user development as the future of software development (Fischer & Giaccardi, 2004; Fischer, Ye, Sutcliffe, & Mehandjiev, 2004). IT consultants also recommend “do it your-self”, DIY, approach for SMEs (Hason, 2001). It is necessary to explore a suitable end-user development approach that can be used by SMEs.

The scope of this thesis is limited to formulating a suitable end-user development approach which can help end-user developers to develop and maintain the Web applications. The findings of the research can help towards developing innovative end-user development environments and tools that can be used by SMEs.
1.5 The Research Questions and Objectives

The literature on end-user development showed that although end-user development could bring many benefits to organisations, very little effort has been taken to adopt the DIY approach for business users specifically SMEs. Therefore, the researcher saw a void in the end-user development area, namely an approach suitable for business users to develop different types of Web applications to IT enable their businesses processes.

The work in this thesis addresses the following research question:

“How to support SME end-users to develop and/or maintain the Web applications to support their business processes?”

This main research question can be divided into a set of sub questions.

- What are the different types of Web applications SME end-users require?
- What are the capabilities of end-users in relation to developing/or maintaining Web applications?
- How do the end-users conceptualise the Web applications?
- Based on the findings for the above three sub research questions, what are the requirements for a suitable end-user development approach for SMEs?
- How to create Web applications to support the identified requirements?

1.5.1 What are the different types of Web applications SME end-users require?

SMEs need different types of Web applications to support their business processes. The Web applications required by SMEs are range from simple Web sites to eCommerce
applications. For example, at the beginning they may want Web sites to have a Web presence for their companies or they may want a customer relation management system to keep track of their clients. They may later want to extend their business on the Internet to an online business. Therefore, it is important to identify the different types of Web applications required by SMEs. This will help to understand the specific support needed by SMEs to successfully develop the Web applications they require to use in their businesses.

1.5.2 What are the capabilities of end-users in relation to developing/ or maintaining Web applications?

Once the different types of Web applications are identified then it is necessary to identify the end-users' capabilities in developing and maintaining those Web applications. This reveals their ability to use implementation domain concepts and tools to develop Web applications. This knowledge could then be used to formulate suitable end-user development environments and tools. This will help SME end-users to use the tools to develop/ maintain Web applications successfully.

1.5.3 How do the end-users conceptualise the Web applications?

It is necessary to analyse how end-users conceptualise Web applications. If SME end-users use the DIY approach in Web application development, then they need to be competent in using application domain concepts. This motivates research to explore how end-users specify Web applications. Further, if the analysis of the capabilities of end-users relating to the implementation domain concepts reveals that end-users have
difficulty in using those concepts, then the conceptual model may be investigated as the input to the end-user development approach.

1.5.4 Based on the findings for the above three sub research questions, what are the requirements for a suitable end-user development approach for SMEs?

The findings for the above three sub questions form the requirements for a suitable end-user development approach that can be used by SMEs. Requirements are derived in three areas: 1) categorisation of Web applications that the development tools should support; 2) the user interfaces of the end-user development tools; 3) the functionality of the end-user development tools.

1.5.5 How to create Web applications based on the identified requirements?

The proposed end-user development approach should cater to the requirements established in 1.5.4. It should support the life cycle of Web applications in both the application domain and the implementation domain. It should support evolution of different types of Web applications. It should use the conceptual level of Web applications as the input from the users. It should further use the end-user development concepts to improve the usability.
1.6 Research Contributions

This thesis contributes significantly to the knowledge of end-user development and Web engineering. Table 1.1 summarises the contributions from the research.

<table>
<thead>
<tr>
<th></th>
<th>Contributions</th>
<th>Contribution appears in the following Chapters</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A set of requirements for an end-user development approach suitable for business users.</td>
<td>Chapter 2: Related work;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Chapter 3: Identifying the Requirements to facilitate End-user Development;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Chapter 4: Conceptual Model of Different Types of Web Applications;</td>
</tr>
<tr>
<td>2</td>
<td>Set of aspects to specify the different types of Web applications required by business users at a conceptual level.</td>
<td>Chapter 4: Conceptual Model of Different Types of Web Applications;</td>
</tr>
<tr>
<td>3</td>
<td>The meta-design paradigm based on the meta-model to support end-user development of Web applications.</td>
<td>Chapter 5: Meta-design Paradigm and the Meta-model;</td>
</tr>
<tr>
<td>4</td>
<td>Strategies for successful use of meta-design paradigm.</td>
<td>Chapter 6: Evaluation of the Meta-model based Meta-design Paradigm;</td>
</tr>
</tbody>
</table>
1.7 Research Approach

Two complimentary but distinct paradigms characterise the research in the Information Systems discipline: behavioural science and design science (Hevner, March, Park, & Ram, 2004; March & Smith, 1995). The behavioural science paradigm seeks to develop and justify theories that explain or predict human or organisational behaviour (Hevner, March, Park, & Ram, 2004). The design science paradigm, which is a problem-solving paradigm, seeks to create innovation artefacts (Hevner, March, Park, & Ram, 2004). The research problem shall dictate the approach and methods to be used (Ollkonen, 1993; Singleton & Straits, 1999). The research question addressed by this thesis is “How to support SME end-users to develop and / or maintain Web applications to support their business processes?”. The characteristics of the research question imply the research is in the design science paradigm since it seeks to create innovations that define practices and technical capabilities which the development, management, and use of Information Systems can effectively and efficiently accomplish.

The research used a constructive research process which incorporated several research strategies: survey, case study and action research cycles within different elements of the main research process. Constructive research is an established method that has been used in management accounting, information systems, computer science and medical domains (Kasanen, Lukka, & Siitonen, 1993; Lukka, 2003). Constructivism creates new realities to solve explicit problems which have both practical and theoretical components (Guba & Lincoln, 1998). The Constructs are all human artefacts such as models, diagrams, plans, framework, or system designs. The constructs produced in this research can be used to help SME end-users to develop and maintain the Web applications required to support their business processes.
The core features of constructive research include:

- Focus on real-world problems that could be solved in practice.
- An innovative construction meant to solve the identified real-world problem.
- Attempts to implement the developed construct and tests for its practical applicability.
- Very close involvement and co-operation between the researcher and practitioners in a team-like manner.
- Links to prior theoretical knowledge.
- Reflection of empirical findings back to theory.

The main elements of the constructive research process are shown in Figure 1.3. Figure 1.4 illustrates the overall constructive research process followed in this research. The next sections briefly describe the key elements of the research strategy taken.

Figure 1.3: Elements of Constructive Research (Lukka, 2003, p. 246)
Research Question

How to support SME end-users to develop and/or maintain the Web applications to support their business processes?

Practical Relevance

SME study: 52% from the 17 SMEs willing to involve in EUD
OCAS project: End-users participate in the Dev. tasks increase the efficiency.

Construction

Requirements for end-user development approach suitable for business end-users

Conceptual Model for different types of Web applications

Meta-design paradigm supported by meta-model

Practical Functioning

Strategies for successful end-user development approach

Research Outcome

Meta-model based
Meta-design paradigm to develop Web applications

Theoretical Contribution

• Conceptual model of Web applications
• Meta-model for web applications

Figure 1.4: Overall research process
1.7.1 Practical relevance

The research process began with investigating the practical relevance of end-user development to SMEs. The first study consisted of a survey followed by a hands-on workshop with 24 Small to Medium Enterprises (SMEs). The second study on end-user involvement was carried out in a large scale project. These studies made a major contribution towards the formulation of requirements for an end-user development approach suitable for SMEs.

1.7.2 Connection to prior theory

The general research area was identified, and gradually narrowed down to the research problem. The research is focused on the development of simple one off applications that can help SMEs. Therefore, the literature survey is narrowed down to the end-user issues, models and approaches relate to development of Web applications. Consideration of the research problem led to a search of the available literature on the three related domains: end-user development, psychology of programming, and Web engineering. A gap in relation to the research question was identified as follows: “there are limited end-user development approaches for developments of different types of Web applications which are required by SMEs to IT enable their businesses.” The findings from this research contribute to develop research areas relating to Web engineering and end-user development.

1.7.3 Construction

Constructions, in general terms, refer to entities which produce solutions to explicit problems. The constructs in this research solve the research question- “How to support SME end-users to successfully develop and /or maintain Web applications?” First, the requirements of a suitable end-user development approach for SMEs are formulated by
identifying the different types of Web applications required by SMEs and looking at their capabilities in relation to developing and maintaining Web applications, and how they are conceptualising the Web applications. Then the aspects at conceptual level of different types of Web applications required to model the Web applications are identified. Next a meta-model based meta-design paradigm to develop Web applications is established. The meta-model is created based on web application patterns. It is used to instantiate different types of Web applications by specifying the values for the aspects of the Web application at the conceptual level. The meta-design paradigm was implemented using component based development and deployment shell (CBEADS). The framework was extended with a Content Management System (CMS) and Workflow management tool which can capture the specifications of different types of Web applications at the conceptual level.

1.7.4 Practical functioning of the solution

The meta-design paradigm based on the meta-model of Web applications is practically evaluated using a series of qualitative studies. A group of SMEs and students participated in these studies. These evaluations took place during the development of the constructs. For example, the study with SMEs happened during the early stage of the research. Therefore, these findings shaped the constructs. Even though the meta-design paradigm is formulated to help SMEs, it is a generic development approach that could be used by both novice developers and end-user developers to develop Web applications. The last study with students demonstrates how it can be used in a class room to help students gain an understanding of Web application development.

The studies establish strategies for the success of the meta-design paradigm such as characteristics of individuals, facilitation, and Infrastructure. In other words, if end-users
are given the required support during the initial development phase the meta-design paradigm can be used in further development and maintenance of the Web applications.

1.7.5 Theoretical contribution

Identifying and analysing the theoretical contribution is an inevitable and crucial phase in constructive research (Lukka, 2003). The major constructs in this research are: the conceptual model of Web applications, the meta-design paradigm and the meta-model. These constructs were presented to the research community at several international conferences (see page v for list of publications). These research outcomes can be beneficial to developing ‘Do it Yourself’ in web application development. The research provides a new way of achieving end-user development and opens it up for further analysis. The next section explains the different research strategies used in the work presented in this thesis.

1.7.6 Research strategies

Table 1.2 summarises the different research methods used in the different studies included in the research process presented in this thesis.

<table>
<thead>
<tr>
<th>Research Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Survey Research</td>
<td>Surveys are used to describe attitudes, opinions, behaviours or characteristics of a considerable sample (Fowler, 2002). Surveys can be administered at a moment in time over a cross section to describe current practice or to evaluate a program or activity in which the participants...</td>
</tr>
<tr>
<td>Case Study Research</td>
<td>have been involved. Also a survey over a length of time with the same population can be used to find changes of opinion or to identify trends.</td>
</tr>
<tr>
<td>Action Research</td>
<td>Case study is used to develop detailed intensive knowledge about a single case or a small number of related cases (Robson, 1993). Case studies can be used to explore existing theories (Blaikie, 2007). They can be used to answer what and how questions as well as why questions (Robson, 1993).</td>
</tr>
<tr>
<td>Action Research</td>
<td>Action research serves dual goals of both improving the situation being studied and generating relevant knowledge (Kazanis, 2004). The research focus is on action, positively influencing the situation (Marsick &amp; Watkins, 1997).</td>
</tr>
</tbody>
</table>

In this research, a survey was administered to identify the attitudes, and opinions as a part of a feasibility study for EUD with a group of Small to Medium Businesses (SMEs) in the Western Sydney region. This survey helped to evaluate the practical relevance of end-user development for SMEs and to identify the capabilities of SME end-users relating to Web application development. The findings help to establish the requirements for a suitable end-user development approach. Surveys were also used during the action research cycle and case study to explore the skills, knowledge and opinions of students. These surveys helped to establish the practical functioning of the constructs.

In this research, an exploratory case study of end-users involved in development activities was conducted to identify the perceived benefits and drawbacks of end-user development for the employees within an organisation. This study along with the SME survey helps to establish the practical relevance of end-user development as a solution.
Case study research also can be used to answer the question “What are the different aspects of Web conceptual model?” Case studies on conceptual models of different types of Web applications are investigated to determine application domain concepts of Web applications. Ethnographic studies are used in fieldwork to identify the perception of participants with their interactions and action.

Case studies also can be used to evaluate the constructs. Two case studies and an action research cycle were performed with a group of users to demonstrate the meta-design paradigm. These studies help to establish the strategies for the successful application of meta-design paradigm based on meta-model.

1.8 Thesis Overview

This thesis consists of seven chapters. The remaining chapters are structured as given in Table 1.3. It consists of a series of integrated studies on End-user development in Web domain.

Table 1.3: Summary of Chapters

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Summary Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chapter 2: Related Work</td>
<td>This chapter discusses the related work in web engineering, end-user development and psychology of programming. The chapter concludes by identifying the gaps in relation to end-user development of web applications related to SMEs.</td>
</tr>
<tr>
<td>Chapter 3: Identifying the Requirements to facilitate End-user</td>
<td>Chapter 3 explains two studies conducted to demonstrate the practical relevance of the research question. These studies identify the benefits and issues in end-user development related</td>
</tr>
<tr>
<td>Chapter 4: Conceptual Model of Different Types of Web Applications</td>
<td>Chapter 4 presents the aspects of different types of Web applications. Literature on the conceptual model of Web applications was reviewed to identify a set of aspects required to model different types of Web applications and three case studies on modelling different types of Web applications at the conceptual level are presented followed by a study of end-user specifications for the usage of these aspects.</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>Chapter 5: Meta-Design Paradigm and Meta-model</td>
<td>This chapter first explains the meta-design paradigm based on the meta-model. The meta-model of Web applications is constructed to support different types of Web applications which can map the conceptual model aspects identified in chapter 4 to the implementation. Then the meta-model of different types of Web applications is explained. This chapter also presents the different complexity levels of the framework, which allow end-users to identify the different tasks in which they can participate. This chapter also includes the theoretical evaluation of the meta-design paradigm.</td>
</tr>
<tr>
<td>Chapter 6: Evaluation of Meta-model based Meta-design Paradigm</td>
<td>Chapter 6 presents three qualitative studies conducted to practically evaluate the meta-design paradigm. The first study was a case study conducted with SMEs. The second case study was conducted with a group of business degree students. The third study is an action research cycle conducted with a group of students following a Web engineering course.</td>
</tr>
<tr>
<td>Chapter 7: Conclusion and Future Work</td>
<td>concludes with the presentation of strategies to be used with proposed meta-design paradigm based on meta-model.</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Chapter 7 provides the conclusion of the study by summarising the findings of the research including the limitations of this research and possible future directions.</td>
<td></td>
</tr>
</tbody>
</table>

1.9 Conclusion

In summary, this Chapter has addressed the background towards the end-user development process for end-user developers in the Web context and the importance of effective new models and frameworks to end-user Web application development. The aim and objectives of the research, research problem and scope, research questions and contribution are also included, and finally, an overview of the Chapters is given.

Parts of the research from this thesis have been published in refereed conferences and journals (see page v for details).
2 Related Work

2.1 Introduction

Chapter 1 outlined the research problem this thesis attempts to answer: “How to support SME end-users to develop and / or maintain Web applications to support their business processes”. This chapter reviews the existing literature related to the research question. The research disciplines related to end-user development in the Web domain is called “EUD Web” (Rode, 2005, p. 35). EUD Web is a cross section of three domains: Web engineering, end-user development and psychology of programming as shown in Figure 2.1 (Rode, 2005). Figure 2.1 also includes the specific areas investigated in each of the three domains related to the work presented in this thesis.

![Figure 2.1: Related Areas adapted from Rode (2005)](image)
Web engineering and end-user development are two complementary domains focused on building better tools, process, and technologies to support development and maintenance of Web applications. Research in the domain of Web engineering focuses on Web development professionals and developing tools, technologies and processes to help them create quality Web applications to meet their customer needs. On the other hand the researchers in end-user development focus on empowering end-users to autonomously create or modify software artefacts. Furthermore, the domain of psychology of programming focuses on improving the usability of programming languages and development environments by better anticipating human needs and evaluating the effectiveness of design solutions (Blackwell, 2006).

This chapter reviews the literature related to the research presented in this thesis in order to identify a gap in the research area. First some key terminologies used in this research are explained. Then related work in three areas, end-user development, psychology of programming and Web engineering are discussed. Finally, the chapter is concluded by highlighting the identified gap.
2.2 Key Terminologies

This section presents a set of terminologies that form the foundation for this research. These terms are End-user, End-user Development, End-user Developer, Meta-design, SMEs, Web Application, Conceptual Model, and Meta-model. Definitions of these terms and their usage in the context of this work are presented here.

2.2.1 End-user

End-user is defined as “A user of an application program”. Typically, the term means that the person is not a computer programmer. A person who uses a computer as part of daily life or daily work, but is not interested in computers (Cypher, 1993). Costabile et al. (2007) use the term “end-user” to denote experts in a specific discipline (e.g., medicine, geology), who, in general, are not expert in computer science but use computer systems for their daily work activities.

In the context of this research, end-users are business people who use the Web applications to help them in their work.

2.2.2 End-user development

At the first EUD-net workshop ("EUD-Net", 2002) held in Pisa, End-User Development (EUD) was defined as a set of activities or techniques that allow people, who are non professional developers, at some point to create or modify a software artifact (M. F. Costabile, Fogli, Fresta, Mussio, & Piccinno, 2003; Paterno, 2003). This definition was slightly modified by Paterno, Klann and Wulf (2003) to a set of methods, techniques, and tools that allow users of software systems, who are acting as non-professional software developers, at some point to create or modify software artefacts. The aim of this modification was to highlight that non-professional developer is a role
rather than a classification. Recently, Costabile, Fogli, Mussio, and Piccinno (2007) have refined the definition of EUD focused on co-evolution as to the set of methods, techniques, and tools that allow end-users to create or modify the interactive system whenever necessary and that support the continuous coevolution of the system and its users. The term EUD indicates the active participation of end-users in the software development process; this can range from providing information about requirements, use cases, and tasks, including participatory design, to activities such as customization, tailoring, and co-evolution (M. Costabile, F., Fogli, Mussio, & Piccinno, 2007).

Some other terms such as end-user programming, end-user computing, user development of applications also have been used in the literature to describe the activities that result in program or computer artifact creation or modification. For example, end-user computing is described as “the adoption and use of information technology by people outside the information system department, to develop software applications in support of organizational tasks (Brancheau & Brown, 1993). When end-users, who have not necessarily been taught how to write code in conventional programming languages, write computer programs, it is defined as end-user programming (EUP) (Cypher, 1993). McGill (2002) has used the term “user development of applications” to explain the same concept and defined it as “The use of Information technology by personnel outside the IS department to develop software applications to support organizational tasks”.

In the context of this thesis, end-user development is used as a set of activities or techniques that allow users of the system, who are non professional developers, to create or modify a software artefact at some point.
2.2.3 End-user developer

End-user developer is someone who develops an application to support his/her work or work of other end-users (McGill, 2002). Sometimes in the literature, the term “end-user programmer” has been used instead of end-user developer. End-user programmers are defined as the people who write programs, but not as their primary job function. Instead, they must write programs in support of achieving their main goal, which is something else, such as accounting, designing a web page, doing office work, scientific research, entertainment, etc. (Myers, Ko, & Burnett, 2006).

In the context of this research, end-user developers are business people who do not possess formal education in Information technology, but are willing to develop or modify the Web applications to help them in their work.

2.2.4 Meta-design

The term meta-design has been used from 1980s in different fields such as information technology, digital networks and nanotechnologies (Giaccardi, 2003). Therefore, the term meta-design is defined and used in many different ways.

Youngblood (1984), defined meta-design in the context of the arts in the emerging telecommunication infrastructure as a strategy that deals with the creation of context, rather than content. From Youngblood’s point of view, meta-design is a model of integrating both technological and social systems in order to create an environment, which allows people to initiate creative conversations, and control the context of their cultural and aesthetic production.

Virlio (1995) defines meta-design as a process of adaptation to electronic media, a neurological form of design that is directed to shape the perceptual and cognitive
systems by information processing, and to colonize and reorganise the organics according to the model of intelligent machines.

De Kerckhove (1997, p. 10) defined meta-design as follows:

“the design of tools, parameters, and operating conditions that allow the end-user to take charge of the final design”.

According to De Kerckhove, the three conditions supported by meta-design are interactivity (physical linking of people or communication based industries), hypertextuality (linking of content or knowledge based industries), and webness or connectedness (i.e. mental linking of people or the industries in the network) (De Kerckhove, 1997). It is regarded as an instrument for mass customisation.

Soddu (1998) defined the term meta-design as the design of a generative principle, which embeds the idea of designers as producers of an idea-product that defines the paradigm and algorithms of transformation.

Fischer, G. et al. has defined meta-design in the context of end-user development as follows:

“Meta-design characterizes objectives, techniques, and processes for creating new media and environments allowing ‘owners of problems’ (that is, end-users) to act as designers. A fundamental objective of meta-design is to create socio-technical environments that empower users to engage actively in the continuous development of systems rather than being restricted to the use of existing systems (Fischer, Ye, Sutcliffe, & Mehandjiev, 2004, p. 35)".
Costabile et al. define meta-design as a technique, which provides the stakeholders in the design team with suitable languages and tools to favor their personal and common reasoning about, and collaboration with the development of interactive software systems that support user work (M. Costabile, F., Fogli, Mussio, & Piccinno, 2005a).

In the context of this work the meta-design is a paradigm that provides process, activities, and techniques to support end-users to develop/or maintain the systems.

2.2.5 Small to Medium Enterprises (SMEs)

Different countries use different parameters to define SMEs. Some use the number of persons employed, amount of capital invested, amount of turnover or nature of the business (Gamage, 2003). The EU Commission established a definition for SME based on four criteria in order to unify the large number of definitions used by different institutions of the EU-member countries. These criteria are: number of employees, turnover, total balance sheet and independence. From the number of variables that could be used, some have settled on using the number of persons employed as the base measurement.

### Table 2.1: Definition of Terms: SMEs and MEs (adapted from European Union, 2003 (Gamage, 2003))

<table>
<thead>
<tr>
<th>Type of Enterprise</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Micro Enterprises</td>
<td>Within SMEs category, micro enterprises are enterprises with fewer than 10 employees</td>
</tr>
<tr>
<td>Small Enterprises</td>
<td>Small enterprises have between 10 and 49 employees</td>
</tr>
<tr>
<td>Medium sized enterprises</td>
<td>Medium sized enterprises have fewer than 250 employees</td>
</tr>
</tbody>
</table>
In the context of this research the term SME is used to denote the Small to Medium Enterprises where the number of employees is less than 250.

### 2.2.6 Web application

The World Wide Web, which emerged as document-centric Web sites, has now evolved into different types of Web applications such as interactive and workflow-based Web applications as well as portals (Schauerhuber, 2007). Baresi, Garzotto and Paolini (2000) have proposed a broader definition for Web application as an extension to information systems or simple Web sites:

1. A Web application can be defined as an extension to the notion of a traditional information system, taking into account the need to incorporate navigation and complex information structures;

2. A Web application can be defined as an extension to a traditional Web site, taking into account the need for incorporating various kinds of application operations.

Kappel et al. narrows down the definition of Web application as follows:

“A Web application is a software system based on technologies and standards of the WorldWideWeb Consortium (W3C) that provides Web specific resources such as content and services through a user interface, the Web browser” (Kappel, Proll, Reich, & Retschitzegger, 2006, p. 2).

In the context of this research Web applications are software systems based on Web technologies that can be accessed through Web browsers.
2.2.7 Conceptual model

A conceptual model defines the conceptual schemas, which express the organization of the application at a high level of abstraction, independently from implementation details (Brambilla, Comai, Fratenali, & Matera, 2007). A conceptual model can be used to verify whether application domain requirements have been properly understood (A. Ginige, 2008). Johnson and Henderson (2002) define the conceptual model as a high-level description of how a system is organized and operates which describes 1) the major design metaphors and analogies employed in the design, if any; 2) the concepts the system exposes to users, including the task-domain data objects users create and manipulate, their attributes, and the operations that can be performed on them; 3) the relationships between these concepts 4) the mappings between the concepts; and 5) the task-domain the system is designed to support. Dori (2003) describes the conceptual model as the complete and unambiguous representation of a system under development.

In the context of this research, the conceptual model is described as a model of application requirement which is a solution to a problem which the system is going to solve. The conceptual model consists of concepts, which users use to specify the application.

2.2.8 Meta-model

Pidcock, W. (2003) defines the term meta-model as follows:

“A meta-model is an explicit model of the constructs and rules needed to build specific models within a domain of interest.”

In other words, a meta-model is a model for the information that needs to be expressed during (software) modelling (Geisler, Klar, & Pons, 1998).
Koch, N. et al. (2008) have more specifically defined a meta-model as follows:

“A meta-model is a precise definition of the elements of a modelling language, their relationships, and the well-formedness rules needed for creating syntactically correct models (Koch, Knapp, Zhang, & Baumeister, 2008, p. 161).”

Milli, Pachet, Benyahia and Eddy (1995) have identified three dimensions of metamodelling:

1. Metamodelling as the modelling of a modelling/representation language.
2. Metamodelling as the multiple instantiation levels of application knowledge.
3. Metamodelling as the modelling of information about how to use and manipulate application models.

Metamodelling can also be defined as a semantic model for a family of related domain models (Amatriain, 2005). In this dimension, metamodelling becomes very much related to ontological engineering. Ontology is the explicit representation of domain concepts. Ontology defines the vocabulary of the problem domain and restricts the way that these terms may be combined to model the domain.

Therefore, in the context of this research a meta-model is defined as an abstract model of an application, which can be used to instantiate the application. User can specify the application he/she wants by defining the attribute values of the meta-model. Then these values can be used to generate the required application.

2.3 End-user development

The focus in creating software systems has shifted from ‘easy-to-use’ to ‘easy-to-develop’ (Lieberman, Paterno, Klann, & Wulf, 2006; Paterno, 2003). Researchers have come up with different concepts and activities that can be used to help end-user
developers. This section, explains the managerial and social perspectives of EUD which establish the practical relevance of the research question, end-user development concepts which are used in this research and end-user development activities.

### 2.3.1 End-user Development - managerial and social perspectives

There is some research investigating the managerial and social perspectives of EUD (Mehandjieva, Sutcliffe, & Lee, 2006). One of the early researches conducted by Fischer, Ye, Sutcliffe and Mehandjieva (2004) summarised the relations between social and managerial issues in EUD as given in Figure 2.2. Their framework was built mainly based on the literature (Brancheau & Brown, 1993; Powell & Moore, 2002) on managerial and social perspectives of the End-user Computing (EUC) discipline. Branchuau and Brown defined EUC as follows.

“The adoption and use of information technology by people outside the information system department, to develop software applications in support of organizational tasks (Brancheau & Brown, 1993, p. 439). ”

The identified user motivations include empowerment from being able to complete the job more effectively, speed of development, flexibility and local control, eliminating the miscommunications of requirements to software developers. User motivation should be encouraged during the early stage of adoption by management support, training, and task forces to spread best practice and expertise. The benefits should offset user costs such as selecting appropriate technology, installing and learning it, programming and debugging. EUD can be beneficial in less complex, less critical and rapidly evolving contexts. Management issues include risks associated with EUD, such as reliability, accuracy, and security. Critical success factors for EUD depend on the domain. For
example, in the scientific and engineering domains where the end-user motivation is high and the managerial influence is low, the success is simply a matter of users taking the development into their hands. However, in most business domains training and technical and management support is vital for the success of EUD. Technology should provide easy integration with other information systems and optimized support for EUD tasks.

![Diagram](image)

**Figure 2.2: Managerial and Social Perspectives (Adapted from (Fischer, Ye, Sutcliffe, & Mehandjiev, 2004))**

Mehandjiev, Sutcliffe and Lee (2006) have conducted a pilot study to investigate the implications of end-user development at social and organizational settings. The pilot study combined a survey and group discussions which attempted to capture the attitudes, opinions and experiences related to EUD within organisations. They have concluded that success of EUD depends on motivating end-users to invest in learning new tools and conversely, on minimising the costs of learning and operation.

Rosson, Ballin and Heather (2004) have summarised the findings from a study of twelve informal Web developers about how they came to doing Web development, their projects, tools used, issues and concerns they had with development tools as follows:
“EUD activities are situated in a collaborative context in which they depend on colleagues for content, expert advice, and testing. Their choice of tools was often based on organizational issues such as cost or who else was using the tool, rather than their own preferences or analysis of tools available. They learned new skills in an informal and as-needed fashion, often by tracking down and adapting or modelling the examples of others (Rosson, Ballin, Rode, & Toward, 2005, p. 523)”.

The early research on managerial and social perspectives of end-user development was mainly focused end-user developers within large organisations. Some researchers attempted to extend this research into informal web developers. However, the managerial and social perspectives of end-user development for SMEs have not been evaluated in the literature.

2.3.2 End-user development concepts

2.3.2.1 Meta-design

The meta-design paradigm in the context of end-user development was conceptualised to address the need to accommodate evolution in information systems. (Fischer & Scharff, 2000; Fischer, Ye, Sutcliffe, & Mehandjiev, 2004). They have specified meta-design as a methodology characterised by activities, processes, and objectives focused on technical infrastructure, learning environment and a socio technical environment. A technical infrastructure that supports meta-design should be an open system where the users are able to create and modify the system according to the evolving needs. Then the users of the system can become in-charge of their systems (Fischer & Scharff, 1998). An approach to realise the meta-design paradigm should be developed focused on a
specific domain (Fischer & Scharff, 2000; Fischer, Ye, Sutcliffe, & Mehandjiev, 2004). Meta-design tools should allow the end-users to learn the activities required to modify the system according to evolving needs. Tools should allow users to use them for the accomplishment of a purpose in hand (Illich, 1973). Natural development and gentle slope concepts which are presented in the next two sections can help to create a suitable learning environment for end-users. The development of a socio technical environment is also important for the success of the meta-design paradigm. A meta-design paradigm encourages participatory development (detailed in section 2.3.2.4). This can support design communities and emergence of the power users (Nardi, 1993).

The SER process model, the seeding, evolutionary growth, and reseeding, extends the traditional development process to include a process in which users become co-designers not only during the design time, but throughout the whole existence of the system (Fischer & Giaccardi, 2005; Fischer & Ostwald, 2002). The SER process model includes two stages of the design process: design time and use time. In this approach it is proposed to under-design the system (i.e. to create an environment that allows owners of the problem to create a solution rather than giving them a solution to use) when creating the systems at the design stage. This is called the ‘seed’. In the evolutionary growth phase users are allowed to design and modify the seed according to their requirements. In this phase the growth is decentralized. Therefore in the reseeding phase the information and artifacts need to be organised, formalised, and generalised to create an effective and efficient system.

The Software Shaping Workshop (SSW) (M. Costabile, F. & et.al., 2003; M. Costabile, F., Fogli, Mussio, & Piccinno, 2005b) methodology is a meta-design approach to develop interactive systems. SSW exploits the metaphor of the artisan workshop, where
an artisan finds all and only the tools necessary to carry out his/her activities. The development team including designers and end-user representatives are supported in their reasoning on software design and development by software environments tailored to their needs, notations and experience. They can also exchange among them and test the results of these activities to converge to a common design.

2.3.2.2 Natural development

Natural development implies that people should be able to work through familiar and immediately understandable representations that allow them to easily express relevant concepts, and thereby create or modify applications (Berti, Paterno, & Santoro, 2004; Paterno, 2003). This approach minimises the “cognitive distance” between user’s intentions and the language specification (Green, 1989). When the programming world is closer to the problem world it would be easier to develop a solution to the problem (Green & Petre, 1996). In other words, if the end-users were provided with development tools and framework which take the primitives close to the aspects in the conceptual domain, they may be able to develop Web applications easily. Rode in his research has identified the need for hiding the implementation domain aspects such as database management, authentication, authorisation in end-user development tools to match to the mental model of end-users on Web applications (Rode, 2005).

2.3.2.3 The concept of gentle slope

The concept of “gentle slope,” mean that “to modify a computer application through its user interface end-users should only need to increase their knowledge by an amount proportional to the complexity of the modification” (MacLean, Carter, & Moran, 1990). There should be a reasonable trade off between ease-of-use and functional complexity and avoidance of big steps in end-user development environments (M. Costabile, F.,
Fogli, Mussio, & Piccinno, 2007). Myers, B. et al. have envisioned that the way to achieve end-user development is development of an ‘ideal gentle slope system’ (Myers, Ko, & Burnett, 2006). However, in their reviews on the progress of end-user development tools and technologies they have reported that such a system has not yet been developed.

### 2.3.2.4 Participatory development

Participatory approaches exploit techniques that are derived from social science that support communication and collaboration within the interdisciplinary team to context analyses, iterative prototyping and requirement specification (Johansson, 1999). Participatory programming goes beyond participatory design and allows end-users and software engineers to collaborate in developing and tailoring software tools (M. Costabile, F., Fogli, Mussio, & Piccinno, 2007). Software Shaping Workshop (SSW) allows its stakeholders to participate in the design process through interpreting and experimenting with the workshop owned by them (M. F. Costabile, Fogli, Fresta, Mussio, & Piccinno, 2003). Gen-Build (Baron & Girard, 2001) provides an environment for domain experts and end-users to collaborate in developing interactive stand alone applications.

### 2.3.3 End-user development activities

Researchers have come up with a range of end-user development activities to help end-user developers to create or modify software artefacts. Costabile et al. (2003) group EUD activities into two broader categories: 1) parameterisation-choose among alternate behaviours, 2) creation and modifications. Two end-user development activities, parameterisation and annotation belong to the first group. The end-user development
activities in the second group can be classified into programming by example, model based development, and extended annotation or parameterisation. However, even during the creation or modification, parameterisation can be used to some extent. For example, the user can provide an option to select a style sheet for a Web site from a given set of style sheets to create his own Web site. Lieberman, H. et al. (2006) also specify a taxonomy of end-user development activities such as parameterisation, annotation, programming by example, model based development, and extended annotation or parameterisation. This section explains different end-user development activities that can be used in an end-user development framework.

2.3.3.1 Parameterisation or customisation

Parameterisation means that the user can guide the computer program by indicating how to handle several parts of the data in different ways; by selecting different functions to apply to data based on given parameters (M. Costabile, F., Fogli, Letondal, Mussio, & Piccinno, 2003). Further, user can also customise the presentation or behaviour of a computer program by selecting a given set of data (Rode, 2005). These forms of activity confine the flexibility for ease-of-use.

2.3.3.2 Annotation

Annotation is described as users writing comments next to data and results in order to remember what they did, how they obtained their results and how they could reproduce them (M. Costabile, F., Fogli, Letondal, Mussio, & Piccinno, 2003).

2.3.3.3 Incremental programming

Incremental programming is described as changes to the behaviour, or presentation of a small part of a program, such as method in a class (Lieberman, Paterno, Klann, & Wulf, 2006). This can specifically be used to modify predefined high-level functions by editing
the behavioural code (Rode, 2005). Another example is writing a glue code (i.e., a high-level script) to record connections between components (Paterno, Klann, & Wulf, 2003).

2.3.3.4 Program by example

In program by example the user provides the example interaction and the system infers the routine from that (Lieberman, 2000, 2001). Sometimes Programming by Example" (PBE) is also called, "programming by demonstration" which means that the user demonstrates examples to the computer (Lieberman, 2001). With programming by demonstration, the user instructs the system to "watch what I do", and a programming by demonstration system creates generalized programs from the recorded actions (Cypher, Kosbie, & Maultsby, 1993). The disadvantage of programming by example is a possible error-prone generalisation which can quickly become a source of frustration for developers if it is invisible, uncontrollable, or based on inappropriate heuristics (McDaniel, 2001).

2.3.3.5 Model based development

Model-based approaches can help end-user development since they can provide abstractions that allow end-user developers to focus on the main concepts (the abstractions) without being confused by many low-level details (Paterno, 2003). For example, M-Base, a message-flow based tool designed for end-users who are familiar with the domain model, is based on the formula “a domain model = a computation model”. That means the conceptual model becomes the input to the implementation domain. The conceptual model of M-Base is based on a 2-Layer model: 1) message-driven model: a dynamic model expressing the system behaviour; 2) class hierarchy model: a static model expressing both specifications and static relations of objects. The
Software Shaping Workshop (SSW) methodology is another model-based methodology, which refers to a model of the interaction and coevolution processes of related members of a work domain to identify the causes of usability difficulties affecting interactive systems (M. Costabile, F., Fogli, Mussio, & Piccinno, 2007).

2.3.3.6 Extended annotation or parameterisation

Extended annotation or parameterisation is an end-user activity which was introduced to support collaborative development (M. Costabile, F., Fogli, Letondal, Mussio, & Piccinno, 2003; M. Costabile, F., Fogli, Mussio, & Piccinno, 2007). The Software Shaping Workshop (SSW) (M. Costabile, F., Fogli, Mussio, & Piccinno, 2007; M. F. Costabile, Fogli, Fresta, Mussio, & Piccinno, 2003), a collaborative development environment which provides annotation, has been extended for cooperative development by allowing users to identify a new functionality by selecting from a set of modifications other people have carried out and which are stored in a shared repository.

2.3.3.7 Spread sheet paradigm

A spreadsheet paradigm is a computationally powerful paradigm which can express textual values as well as graphics (Burnett & Gottfried, 1998). Spreadsheet formulas are "pull"-oriented: a cell expresses its interest in other cells through references in its formula, and updates cause it to "pull" in new intermediate values for a new computation.

2.3.3.8 Component based development

Won, M. et al. (Won, Stiemerling, & Wulf, 2006) has introduced a customizable approach with two-dimensional and three-dimensional customizable environments based on a component model called Flexibean, which can be used effectively by end
users. Integrity checks are built into the customizable environment, and customizable artifacts can be exchanged through a shared repository.

2.3.4 Conclusions and implications for realising meta-design paradigm

End-user development concepts and end-user development activities discussed in this section can help to realise the meta-design paradigm. The SER process model and SSW are example realisations of the meta-design paradigm in different domains. These can be used in developing a suitable end-user development approach for SME end-users to develop and maintain the Web applications required to support their business processes. Therefore, the end-user development concepts presented in this section are used to compare the existing end-user development approaches in section 2.5.3.

2.4 Psychology of Programming

One objective of psychology research is to apply the understanding of human cognition studies to improve the usability of development tools and environments by better anticipating the needs of the users (Blackwell, 2006). Cognitive scientists have come up with concepts, theories and models to help end-users to accomplish different tasks related to software development and maintenance (Rode, 2005; Shneiderman, 1980). This section, explains the concepts and theories in the psychology of programming related to the work presented in this thesis.

2.4.1 Mental model

In 1943, Kenneth Craik formulated the concept of mental model as follows. Human beings translate external events into internal models and reason by manipulating these symbolic representations. They can translate the resulting symbols back into actions or
recognize a correspondence between them and external events (Craik, 1943). Norman in his book “The Design of Everyday Things” argued that mental models are what people really have in their heads and what guides their use of things (Norman, 1989). Linxiao et al. have assessed the mental model of novice programmers and identified that people with a viable mental model perform well in accomplishing development tasks (Linxiao, John, Marc, & Murray, 2007). In Rode et al.’s work on everyday programmers they found that end-users do not focus on implementation domain concepts such as session management, authentication, authorisation, database management, binding data with interfaces, user input validation (Rode, Rosson, & Perez-Quinones, 2004). Therefore, they argue that end-user developers should support high level concepts that end-users are familiar with such as page, table, and button.

2.4.2 Runtime programming

The objective of runtime programming (Rode & Rosson, 2003) is to make it possible for both to develop and use the application without switching back and forth between programming and runtime modes. The end-user developer alternates between constructing and using the application until he or she gets the application developed according to his or her requirement. End-users receive a true WYSIWYG (What you see is what you get) experience with runtime programming. However distinguishing the execution mode and edit mode will be difficult in this approach and therefore there is still a need to provide a minimal notion of run time mode vs. design mode.

2.4.3 Minimalism

Carroll's studies on computer use have demonstrated that for the most part end-users do not want to “learn” but rather to “produce”, and therefore, are willing to use whatever information or resources are available to help them make sense of a task just enough to
make progress (Carroll, 1990, 1998; Carroll & Mack, 1999; Warren, 2004). This pattern of behavior is also common among end-user developers (Rode, Rosson, & Perez-Quinones, 2004; Rosson, Ballin, & Heather, 2004). The minimalism theory was established to solve these issues with the behaviour of end-users. Minimalism describes movements in various forms of art and design which reduce the subject to its necessary elements ("Minimalism", 2008). Minimalism theory related to the computer domain suggests that (1) all learning tasks should be meaningful and self-contained activities, (2) learners should be given realistic projects as quickly as possible, (3) instruction should permit self-directed reasoning and improvising by increasing the number of active learning activities, (4) training materials and activities should provide for error recognition and recovery and, (5) there should be a close linkage between the training and actual system (Carroll, 1998). Minimalism is proposed as a framework supporting development of effective instruction manuals. This minimalist philosophy also helps in designing an end-user development approach.

2.4.4 Conclusions and implications for realising a meta-design paradigm

Based on the literature on mental models, it can be concluded that for the success of end-user development it is required to provide an end-user development tool which takes a higher level abstract inputs rather than implementation domain concepts. Runtime programming concepts should be used with such tools. Further, the minimalism theory can provide guidelines to meta-design tools.
2.5 EUD Web

Web site development is becoming common amongst end-users to meet both leisure and professional goals as the development tools provide higher level abstractions hiding the low level concepts (Rosson, Sinha, Bhattacharya, & Zhao, 2007). EUD Web research projects such as WebFormulate (Ambler and Leopold 1998), FAR (Burnette, Chekka, & Penday, 2001), DENIM (Newman, Lin et al. 2003), and WebSheets (Wolber, Su, & Chiang, 2002), Click (Rode, Bhardwaj, Pérez-Quiñones, Rosson, & Howarth, 2005), and QEDWiki ("QEDWiki", 2006) have explored specific approaches to end-user development of Web applications. The objective of this literature survey is to investigate what needs to be done to help SME end-users to develop their Web applications to support their businesses. Therefore, it is also required to investigate the type of Web application these different tools can support. This section reviews these EUD Web tools and frameworks supporting Web application development in terms of development activities and types of application.

2.5.1 Different types of Web applications required for SME end-users to support their businesses

End-users require different types of Web applications to support their business processes such as simple Web sites, ecommerce applications. Therefore, the roadmap to eTransform SMEs was used to select the different types of Web applications that required supporting SMEs. Figure 2.3 shows the eTransformation Roadmap (A. Ginige, Murugesan, & Kazanis, 2001) developed by AeIMS Research Group to help Small to Medium Enterprises in the Western Sydney Region to IT enable their business processes. The eTransformation Roadmap is used to evaluate the current position in terms of IT sophistication and provide guidance to move on the path of
eTransformation for the SMEs based on how people, processes and technology interact. It is identified that different types of Web applications and support infrastructures are required in two dimensions namely internal and external process to move to an e-business.

Table 2.2 explains the internal and external processes. This was used as the guideline to select the various types of Web applications sufficient to support SMEs from the different existing Web application categories. The different types of Web applications are defined in Table 2.3. Identified support infrastructure such as e-mail, package software and shared repository is excluded in further discussions.

Figure 2.3: The eTransformation Roadmap adopted from Kazanis (2004)
Table 2.2: External and Internal processes in Roadmap adopted from Kazanis (2004)

<table>
<thead>
<tr>
<th>Ext./Int. Process</th>
<th>Description</th>
<th>Types of Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Website</td>
<td>Display some information of the organization.</td>
<td>Information Centric (A.k.a. Document-centric)</td>
</tr>
<tr>
<td>Interactive Website</td>
<td>Provide up to date product and service catalogue. E.g.: registration forms, customized information presentation, catalogue</td>
<td>Data Intensive</td>
</tr>
<tr>
<td>E-Commerce Site</td>
<td>Has the ability to accept orders and monetary transactions for goods and services. E.g.: online shopping (ordering goods and services), online banking, online airline reservation, online payment of bills</td>
<td>Transactional</td>
</tr>
<tr>
<td>Effective Individual</td>
<td>Technology that enhances a single person’s ability. E.g.: PC and accounting software</td>
<td>Package software</td>
</tr>
<tr>
<td>Effective Team</td>
<td>Technology that allows people to work together. E.g.: eMail, shared drive</td>
<td>Network, eMail, shared repository</td>
</tr>
<tr>
<td>Effective Organisation</td>
<td>Technology that crosses entire organization and enhances a core process of organisation. E.g.: ERP system, manufacturing control system</td>
<td>Workflow intensive</td>
</tr>
</tbody>
</table>
### Table 2.3: Definitions for different types of Web applications

<table>
<thead>
<tr>
<th>Type of Web application</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Information centric (aka: document centric)</td>
<td>A collections of static HTML documents (basically, plain text and images) organised in a hierarchy that offers read-only information based on a set of structured content, navigation patterns, and presentation characteristics designed and stored at the design time (Moreno, Romero, &amp; Vallecillo, 2008).</td>
</tr>
<tr>
<td>Data intensive</td>
<td>Web sites for accessing and maintaining large amounts of structured data, typically stored as records in a database management system (Brambilla, Comai, Fratenali, &amp; Materia, 2008, p. 221).</td>
</tr>
<tr>
<td>Workflow intensive</td>
<td>Web application, which provides support for modeling structured business processes, activity flows, business rules, interactions among actors, roles, and a high-performance infrastructure for data storage (content management). Information is needed not only for the system actors but also for its processes (Kappel, Proll, Reich, &amp; Retschitzegger, 2006; Moreno, Romero, &amp; Vallecillo, 2008).</td>
</tr>
<tr>
<td>Transactional</td>
<td>Web applications, which incorporate support for persistent data store, information location, concurrency control, failure, and configuration management (Moreno, Romero, &amp; Vallecillo, 2008, p. 367).</td>
</tr>
</tbody>
</table>
2.5.2 EUD Web Tools

WebFormulate (Ambler & Leopold, 1998; Leopold, Heimovics, & Palmer, 2002) is an early tool for building Web applications that uses a form-based visual language. It allows developers to construct new computations by referencing other objects via point-and-click. WebFormulate uses a message passing paradigm that reports any changes to an object immediately to all interested objects. The development environment running within the Web browser communicates with the Web server through a hidden HTML frame. However, business logic needs to be coded by the end-user developer.

FAR (Burnette, Chekka, & Penday, 2001) is an online prototype tool which can be used by end-users to develop e-Services. Far has adapted the model of spreadsheet. It models each area in a Web page as a cell or table of cells and operations can be applied over the cells. Users can drag and drop these cell objects to create the pages and specify rules or formulas to apply over cells. Therefore, Far may be a suitable tool to develop calculation intensive applications. But how its expressive power will scale to the design of general Web applications is unclear.

WebSheets (Wolber, Su, & Chiang, 2002) is a tool which can be used to develop dynamic Web content. It consists of a WYSIWYG interface tool for building dynamic Web pages that access and modify databases. Without programming, designers can specify not only the presentation of a page, but the dynamic content as well. This capability is facilitated through a novel application of Programming by Example (PBE), Query by Example (QBE), and spreadsheet formulas within the WYSIWYG HTML editor environment. Web Sheet only supports one to one mapping between database tables and HTML data tables.
DENIM (Newman, Lin, Hong, & Landay, 2003) is a prototypical tool that can help professional and nonprofessional Web developers to model Web sites at different refinement levels (i.e. site map, storyboard, and individual page) in the early stage of development. However, end-user developers have to learn to transform the informal prototype into a final application if they want to create production level Web applications.

FlashLight (Rode & Rosson, 2003) is a prototype tool which covers the programming at run time concept. It's developed using flash and other Web technologies. FlashLight allows users to develop multi-screen Web applications for data collection. i.e. in other words FlashLight caters for interactive application development. Fully functional GUI components can be drag and dropped into the workspace and dynamic behaviour of the application is controlled by defining action rules. FlashLight provides the common set of components like session management and database management needed in Web applications.

Click (Rode, Bhardwaj, Pérez-Quiñones, Rosson, & Howarth, 2005) is a WYSIWYG-based Web application framework which can help end-users to develop data intensive Web applications. In Rode’s end-user development tool for Web applications, Click (Rode, 2005), he has introduced seven complexity layers from customization of an exiting Web application to coding new components in the framework in which the Web application is developed. Layer 1, developers may customize existing Web applications. At Layer 2, developers may use Click’s wizards to create a related set of components. At the next layer, developers can use Click’s form-based user interface to insert new components, customizing the component behaviour through parameterisation. At Layer 4 the developer can manually edit the layout code. The predefined high-level functions
may be modified by editing the behavioural code. At this level, developers have the flexibility to define Boolean conditions of nearly unlimited complexity but are not required to write low-level code. At Layer 6, developers may modify the component-based PRADO framework, which was used to develop Click. At this level developers can also modify Click’s high-level functions or create a new high-level function for use by themselves or other Click users. At the final and most powerful layer, experienced developers can code in PHP. This allows end-users with limited domain knowledge in Web engineering to start developing data-intensive Web applications. Click attempts to match to the end-user mental model of solution domain, but assumes end-users are able to map the problem domain concepts into the solution domain such as data base modelling and authentication.

QEDWiki (Helman & Fertal, 2003) is a browser-based assembly canvas used to create Web applications by assembling widgets (components) called “mashups” in a Web page. Interface objects can bind the software components (or services) made available by content providers. The objective of QEDWiki is to allow end-users to develop the Web pages by dragging and dropping the widgets on to a pallet.

Desk is a Web authoring tool which can be used by end-users to develop dynamic Web content for information centric Web applications (Macias & Castells, 2007; Macias, Puerta, & Castells, 2006). It is based on programming by example technique. DESK identifies the end-user activity by monitoring the user action and infers the knowledge necessary to provide the user with assistance during the authoring process. DESK has improved the ease of use using a WYSIWYG user interface, but is limited in expressive power.
Valderas, Pelechano and Pastor (2006) present an end-user development tool that can be used to develop specific types of Web applications such as e-commerce applications, directories and Web portal. This tool is built upon the OOWS, a model driven development approach to generate Web applications from the conceptual model. The OOWS method has been extended as follows to achieve the end-user development: 1) A tool to capture the description of Web applications in terms of the end-users’ knowledge about the application domain, 2) Then it automatically obtaining an OOWS requirements specification from their description and then generating a Web application prototype and 3) Next a tool to select then most suitable design template from a list of predefined ones for personalising the Web application look and feel. An ontology-based strategy is introduced to give support to end-users throughout the task of developing Web applications.

2.5.3 Conclusions and Implications for realising meta-design paradigm

Table 2.4 categorises the available end-user development tools supporting Web application development based on concepts used, activities supported and type of application. According to the table, most of the tools attempt to match with the end-user mental-model by providing a natural development approach. However, none of the tools supports participatory development and gentle slope while providing a natural development approach, thus not catering for the meta-design paradigm.

Most of the tools support only data intensive Web applications. Denim supports information-centric Web applications. It also shows that end-user development tools have been developed targeting a specific application category. Therefore, if an end-user wants to develop different types of Web applications, then he needs to learn to use
different end-user development tools and environments. The OOWS based EUD tool is an exception to the other existing end-user tools since it can support different types of Web applications. However, for the success of that approach, first, the type of ontology needed to support different types of Web applications has to be completed. Then, the tool needs to be evaluated.
### Table 2.4: Comparison of EUD Web Tools

<table>
<thead>
<tr>
<th>EUD Web Tool</th>
<th>Concepts</th>
<th>Activities</th>
<th>Type of Application</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Natural dev.</td>
<td>Participle dev.</td>
<td>Gentle Slope</td>
</tr>
<tr>
<td>Web Formulate</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Far</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>WebSheet</td>
<td>X</td>
<td></td>
<td>Parameterisation, Programming by Example</td>
</tr>
<tr>
<td>Denim</td>
<td>X</td>
<td>X</td>
<td>Prototype</td>
</tr>
<tr>
<td>Flashlight</td>
<td>X</td>
<td></td>
<td>Parameterisation or Customisation</td>
</tr>
<tr>
<td>Click</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Desk</td>
<td>X</td>
<td>X</td>
<td>Programming by Example</td>
</tr>
<tr>
<td>QEDWiki</td>
<td>X</td>
<td>X</td>
<td>Component based Development</td>
</tr>
<tr>
<td>OOWS EUD tool</td>
<td>X</td>
<td></td>
<td>Model driven Development</td>
</tr>
</tbody>
</table>
2.6 Web Engineering

Web engineering has emerged as an independent branch of software engineering to generate and organise knowledge about Web application development and to apply that knowledge to develop Web applications, or to address new requirements or challenges (Murugesan & A. Ginige, 2005). This section reviews the suitability of existing Web development approaches including Web development tools and frameworks for end-user development by business end-users.

2.6.1 Web development tools

Fraternali (Fraternali, 1999) categorised the existing Web application development tools that support the development of data intensive Web applications into six groups based on homogeneous features called visual editors and site managers, Web enabled hypermedia authoring tools, Web DBPL integration tools, Web form editors, report writers and database publishing wizards, multi paradigm tools and model-driven application generator tools. These tool groups are briefly explained in Table 2.5 with a few examples for each group.

<table>
<thead>
<tr>
<th>Tool Group</th>
<th>Description</th>
<th>Example tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visual editors and site managers</td>
<td>Authoring and site management environments allow writing HTML code and maintaining the pages of a Web site in the file system.</td>
<td>Macromedia</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dreamweaver</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Microsoft FrontPage 2003</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Expression Web Designer (Patton, 2007)</td>
</tr>
<tr>
<td>Tools</td>
<td>Description</td>
<td>Examples</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Web enabled hypermedia tools</td>
<td>Tools supporting the development of offline hypermedia applications, that have extended to support the generation of applications for the Web.</td>
<td>Asymetrix’s Toolbook II Assistant/Instructor Macromedia’s Director</td>
</tr>
<tr>
<td>DBPL integration tools</td>
<td>Tools that can produce dynamic Web pages from information stored in a database by integrating databases and Web technology at the language level.</td>
<td>FileMaker Pro 7</td>
</tr>
<tr>
<td>Web form editors, report writers and database publishing wizards</td>
<td>Rapid Application development (RAD) tools and languages supporting development of forms, reports and publishing.</td>
<td>Microsoft Info Path Quask Form artist</td>
</tr>
<tr>
<td>Multi paradigm tools</td>
<td>Tools integrate solutions from the aforementioned categories into a unified development framework.</td>
<td>Microsoft Visual Web Developer 2005 Ruby on the rails</td>
</tr>
<tr>
<td>model-driven application generator tools</td>
<td>Tools that support automatic generation of the applications using high level abstract models.</td>
<td>Oracle’s Web Development Suite</td>
</tr>
</tbody>
</table>

Some of these tools such as FrontPage or Dreamweaver can help end-user developers since they do not require programming skills as such, but still have a considerable learning curve (Rode, Rosson, & Perez-Quinones, 2004).
Valderas, Pelechano, & Pastor explained the issues with these tools as follows.

“For instance, simple tasks such as implementing the look and feel of the web application become difficult when they have to use HTML flow-based positioning instead of the more intuitive pixel-based positioning. In addition, none of these tools addresses the entire process of web application development since they mainly focus on graphical design tasks (Valderas, Pelechano, & Pastor, 2006, p. 530).”

To help end-users develop Web applications, it is required to reduce expressiveness in favour of increasing ease of use, something that is barely visible in existing commercial Web application development tools today (Macias & Castells, 2007; Rode, Rosson, & Perez-Quinones, 2006).

From these categories of Web tools a multi design paradigm can benefit end-user developers to build different types of Web applications. There are Web development frameworks that can support a multi design paradigm. The suitability of such frameworks to be used in end-user development is discussed in section 2.6.2. Further it is revealed that model driven application generator tools go beyond supporting the lifecycle of the Web applications. The model driven development approaches and meta-model that can be used to generate the Web applications are discussed in section 2.6.3 and section 2.6.4 in detail.
2.6.2 Web development frameworks

A Web application framework is defined as a software framework that is designed to support the development of Web applications and Web services ("Web application framework", 2008). The objective of a framework is to reduce the work required to develop an application by providing support for common activities used in Web development. For example, many frameworks provide libraries that support common activities such as database access, templating frameworks, debugging facility and session management. Code reuse is facilitated by allowing the developers to create their own libraries. A major focus of such frameworks is to improve the productivity of the Web developers using those tools (Helman & Fertal, 2003; Rode, 2005).

Most of the Web application frameworks are based on the MVC design paradigm. Therefore focused on building content (the model), presentation (the view), and navigation (the controller) of a web application. These MVC-based Web application frameworks can generate the necessary code for activities such as business logic, authentication and object mapping. Some of the popular frameworks are: Ruby on rail for Ruby, Catalyst and Maypole for Perl, cake PHP, Prado for PHP, Django and TurboGears for Python.

Vosloo and Kourie (2008) identify a comprehensive set of requirements for the Web application development framework which includes presentation, validation, navigation, session management, back-end integration, form handling, authentication, event handling, concurrency, resource usage and miscellaneous. These requirements were identified by analysing more than 80 available Web development frameworks as a guideline for any new framework. Even though these requirements are equally important
in Web application development for end-user developers these frameworks have not implemented these features focused on end-users. They do not support end-user development concepts such as gentle slope or minimalism. Therefore, the users of these tools have to act at some point as a skilled designers and developers dealing with Web-based languages (or at least with a visual representation of them) and need to learn the formal development concepts (Macias & Castells, 2007).

### 2.6.3 Model driven approaches

Model-driven development approaches are focused on generating the final implementation of a system based on the abstract models of the system (S. Mellor, Clark, & Futgami, 2003; Thomas, 2004). The models are constructed on a high level, and then transformed successively into lower-level models (with increasing detail) until a representation is reached which can be executed (S. J. Mellor & Balcer, 2002). MDA (Model-Driven Architecture) is OMG’s initiative to implement a set of standards and tools based on the model-driven development philosophy (OMG, 2003). In MDA, the Computation Independent Model (CIM) specifies the domain model which captures the domain expertise and explains with domain specific vocabulary. This is transformed to a Platform Independent Model (PIM) which is a system model that does not depend on specific technology. Finally, the PIM is transformed to the PSM in MDA, which is a system model which targets a specific technology. The main advantages of model driven development are quality improvement, interoperability and maintainability.

The model driven development approaches to Web applications can mainly be grouped in to two categories: hypermedia design approach and object oriented design approach (Moreno, Romero, & Vallecillo, 2008). The majority of the design approaches grouped into the hypermedia design approach are based on the E/R model or an extension of it.
Some examples of development approaches grouped into this category include HDM-lite (Fratenali & Paolini, 1998), WebML1 (Ceri, Fratenali, & Bongio, 2000), W2000 (Baresi, Colazzo, & Mainetti, 2005), and WSDM (Paterno, 2000). The approaches such as OOHDM (Schwabe, Rossi, & Barbosa, 1996), UWE (Koch & Kraus, 2002), and OOWS (Pastor, Fons, Pelechano, & Abrahao, 2006) are in the object oriented design approach. In the model driven development approaches to Web applications, first different aspects required to implement Web applications such as content, navigation and presentation are modelled. Then model compilers are used to produce the Web pages and logic from these models.

These model driven development approaches to Web applications are focused on helping developers to build better Web applications in less time rather than helping end-users to develop / expand Web applications. These approaches have the following issues when being considered for end-user development:

- Tools assume expert-knowledge in Web application development (Rode, 2005; Valderas, Pelechano, & Pastor, 2006). For example the models for most of these approaches need to be specified in Unified Modelling Languages (UML) or other proprietary notation where the users need to be comfortable with a set of notations and tools and be able to interpret their requirements in terms of these notations.

- Since these approaches were originally conceived to deal with particular types of Web applications (such as data intensive Web application or information centric Web application), they deal with a fixed set of common aspects such as navigation and presentation (Moreno, Romero, & Vallecillo, 2008).
• Further, as Rode (2005) pointed out, these approaches require considerable planning effort to develop Web applications and have a long feedback loop which makes it difficult for goal driven end-users to develop the Web applications.

However, these model driven development approaches could be improved to support end-user development by extending to support end-user development concepts. One such attempt is the recent extension to OOWS (Valderas, Pelechano, & Pastor, 2006).

2.6.4 Meta-model of Web applications

There are several different meta-models for Web applications based on meta object facility and UML profiles. These meta-models achieve interoperability and standardise the web models. One such model is UWE meta-model (Kraus & Koch, 2003). The objective of UWE meta-model is to provide a common meta-model for the Web application domain, which will support all Web design methodologies., a successor of HDM (Garzotto, Paolini, & Schwabe, 1993), has provided model semantics and transformation rules to achieve consistency between models. Muller et al. (Muller & et.al., 2005) present a model-driven design and development approach with the Netsilion tool. The tool is based on a meta-model specified with MOF 1.4 and the Xion action language. Recently another two meta-models (Schauerhuber, 2006) based on MOF and UML 2 profiles are presented for WebML design methodology with the objective of interoperability.

Moreno et al. (2008) have recently come up with a meta-model based on different types of Web applications. They have captured 13 meta-models organized into three main
packages called user interface, business logic and data. The model is proposed to explicitly model the concerns and specify in a platform-independent manner.

All these meta-models of Web applications are trying to provide a precise definition of the semantics of existing Web models. They support model driven development and generate the Web applications from the models. The main objectives of these approaches are to achieve modularity, portability, reusability, and interoperability. However, it was not clear how these can support inputs at the conceptual level which is a requirement of end-user developers particularly business end-users.

2.6.5 Conclusions and implications for EUD Web development methodology

The tools and methods to Web application development focused on improving the productivity of the developers and quality of the Web applications. Most of the existing Web development tools and frameworks can only support development of specific type of Web applications. Therefore, they lack support for end-user development, particularly for business end-users who are not able to afford the effort required for learning different tools for development or maintenance of the Web applications that they need to support their business processes.

2.7 Conclusion

The review of the literature resulted in identification of the main gaps that form the basis for the research questions detailed in chapter 1. The managerial perspective of end-user development in the context of SMEs has not been studied. Therefore, both their capabilities related to Web application development and the requirements of a suitable
end-user development approach need to be identified. The eTransformtion strategies provide a list of different types of Web applications that SMEs would require to support their business processes. This knowledge can be used to develop meta-design tools which can be used by the SMEs. Further, the reviews of end-user development and the psychology of programming identify a set of concepts that can be used in a Web development approach suitable for business end-users. However, the review of EUD Web shows that there’s a lack of research into development tools and environments that can help to develop different types of Web applications. The analysis of different Web development tools identifies a few tools that can support end-user development of specific type of Web applications. However, the focus of these tools is to improve the productivity of the professional developers. Therefore a ‘gap’ is identified which is, ‘there is limited research into an end-user development approach that can support the development of different types of web applications which is required by SMEs to IT enable their businesses’. The end-user development concepts discussed in sections 2.3.2 and 2.4 were used in building the meta-model which is explained in chapter 5.
3 Identifying the Requirements to Facilitate End-user Development

3.1 Introduction

As mentioned in chapter 1, the objective of the research presented in this thesis is to help SMEs to cTransform their business processes. End-user development is identified as one possible approach to solve the research question. However, according to the findings in chapter 2, there is little research on suitable end-user development environments and tools focusing on business users. This chapter presents two studies on end-user development which aim to identify the requirements of a suitable end-user development approach for business users: 1) Feasibility study with a group of SMEs on end-user development and 2) Exploratory study on end-user participation in Web application development. These two studies further demonstrate the practical relevance of the research question.

The study with SMEs identifies the benefits and issues of end-user development and the capabilities of SMEs related to Web application development. Analysis of the issues and capabilities of SMEs helped to identify the requirements of an end-user development approach suitable for business users particularly for SMEs. The study was conducted
using two questionnaires combined with a hands-on-session. The first questionnaire surveyed the attitudes, opinions, capabilities and experiences of SMEs related to Web application development. A hands-on session on the development of an information centric Web site was conducted. The second questionnaire given at the end of the hands-on-session captured the feedback from SMEs on the tasks they were involved in during the hands-on-session. The participant’s responses to the questionnaires and the task completion rates in the hands-on-session were used to identify their capabilities related to Web application development. Their willingness to participate in Web applications development was also sought. In this survey, 80% of SMEs stated their willingness to participate in end-user development. The biggest issue they face in end-user development is lack of know-how.

The exploratory case study on end-user participation identifies the benefits and issues in end-user development of Web applications from the point of view of the end-users involved in end-user development. The project involved developing an Online Course Approval System (OCAS) for the University of Western Sydney (A. Ginige & J.A. Ginige, 2007). The objective of OCAS was to support the course approval process in the University. During the project, the author observed the end-users’ participation in different development activities such as developing user interfaces, updating state tables, and user management. At the end of the project the end-users who participated in the development were interviewed. The questionnaire used for the interview is given in Appendix I. This exploratory study on end-user participation reveals that end-user involvement reduced the time the taken for development and made it easy to evolve the system to adapt to changing requirements.
This Chapter first presents the “grounded theory” which was used to analyse the data in these studies. Then the study conducted with SMEs is presented. Next the exploratory study on an end-user participation in Web application development is explained. After that, the set of requirements for the end-user development approach and the tools for business users which were derived from the two studies are presented. This is followed by the conclusion to the chapter.

3.2 Grounded Theory

Grounded theory is a qualitative research methodology used to study complex social behaviour from a sociological point of view. In the grounded theory approach, the theory is developed inductively from the data, being therefore generated (or grounded) in a process of continual sampling and analysis of data (Pidgeon, 1996; Strauss & Corbin, 1990).

In grounded theory, the process of analysis starts with “coding” the data. Coding consists of identifying and labelling the key concepts and relevant features of transcript (Seaman, 1999). Carvalho, Scott and Jeffery have explained the process of analysis as follows:

“The researcher searches the data for similarities and diversities, collecting a number of indicators that may point to multiple qualitative aspects of a potentially significant concept. The researcher designates labels to passages that seem relevant to an idea of interest in the study. Then, these labelled passages of the text are searched for patterns, and grouped together. These groups (or categories) are then examined in search of meanings, themes and explanations of the phenomena (Carvalho, Scott, & Jeffery, 2005, p. 115).”
The codes may be derived from the goals for the study, research questions and pre-established variables of interest (Seaman, 1999). On the other hand, researchers may post-form the codes if the study objectives are very open and unfocused. However, in both cases, it is possible to add, delete, modify, merge or subdivide the codes as the study progresses.

The sources of data used for interpretation with the study may be gathered through the use of observations, questionnaires, interviews, books, or documents. The studies presented in this Chapter use the grounded theory to analyse the responses to the questionnaires, and interview data.

3.3 SME Study on EUD

The study conducted with SMEs helps to establish the social and managerial perspectives of EUD in the context of SMEs. This study also identifies the capabilities of SMEs related to Web application development while investigating the practical relevance of end-user development to SMEs. It combines two questionnaires and a hands-on-session to capture required data. The first questionnaire (given in Appendix F) is planned to survey the attitudes, opinions, capabilities and experiences of SMEs related to Web application development. An assignment to develop an information-centric Web application was given during the hands-on-session. Content Management System (CMS tool) was provided to accomplish the assignment tasks. The note used in the session is given in Appendix G. The second questionnaire (shown in Appendix H) given at the end of the hands-on-session captured the difficulty level of the tasks participants had to complete using the CMS application for the assignment as perceived by them. The data gathered in two questionnaires combined with completion rates of the tasks in the
assignment were used to establish the capabilities of SMEs in relation to Web application development.

### 3.3.1 Aim and objectives

The overall aim of the study is to gain an insight into the perceived costs and benefits to SMEs from the adoption of EUD techniques and tools and to identify their capabilities related to Web application development thus to ascertain the practical relevance of EUD.

Therefore, the objectives of the survey are as follows.

**Objective 1:** To identify the perceived benefits of EUD to SMEs (Motivations).

**Objective 2:** To identify the barriers to EUD.

**Objective 3:** To identify the end-users’ capabilities related to Web application development.

**Objective 4:** To analyse the feasibility of EUD.

### 3.3.2 Method

The study targeted SMEs in the Western Sydney region. Data were gathered in two workshops conducted for SMEs by AcIMS Research Group at University of Western Sydney. The first workshop was conducted for the members of the Innovative Technology Network (ITN). The second workshop was organized with the tool makers association (Austool ltd.) as a part of a seminar series at their organization.

The workshop was organized as the last event of a series of workshops on E-Transformation. In the first two workshops, a methodology (roadmap) (A. Ginige, Murugesan, & Kazanis, 2001) that can be used for e-Transformation was introduced.
Also 7Es (Arunatileka & A. Ginige, 2004), a strategy to use in conjunction with the roadmap, which can help SMEs to manage the e-Transformation, was discussed. The topics discussed in these workshops included the purpose of a Web site, how it can help SMEs and what information needs to be in a Web site, good and bad practices in creating Web sites, how to do a navigation structure for a Web site and Web marketing. These two workshops were designed particularly to provide required knowledge about E-Transformation, which is essential to the growth of an organization with Web presence. EUD workshop was conducted after these two workshops. EUD workshop had allocated an hour and a half time.

Questionnaire 1 was given at the beginning of the EUD workshop. Questionnaire 1 consisted of questions to gather some data about respondents including their experience and education related to Information Technology including Web development and attitudes towards EUD. In the grounded theory approach, the questionnaire is treated as a set of generative questions to help to guide research. For example, the questionnaire consists of a semi structured section to establish the current EUD practice among SMEs. This questionnaire is included in Appendix F.

A hands-on-session was conducted after the questionnaire 1. The hands-on-session was structured in two stages. During the first stage, the CMS tool was demonstrated following a brief session on EUD. After that, participants were given the assignment. This assignment covered the tasks of creating and maintaining a Web site given in Table 3.1 using the CMS tool. The end-user development concepts presented in chapter 2 were used in the CMS tool. In the CMS tool, users can develop and maintain Web applications using end-user development activities. The CMS application is explained in section 3.3.4. Additionally, the instructors were there to help them, but the questions
they asked were noted down. Roughly one hour was allocated for the hands-on-tasks. The usage log of the tool was used to identify the completion of each task by the participants.

**Table 3.1: The assignment tasks to create/maintain Web site**

<table>
<thead>
<tr>
<th>Task</th>
<th>Task Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create site</td>
<td>Initiation</td>
</tr>
<tr>
<td>Select templates</td>
<td>Presentation management</td>
</tr>
<tr>
<td>Add page</td>
<td>Content management</td>
</tr>
<tr>
<td>Edit page</td>
<td>Content management</td>
</tr>
<tr>
<td>Upload images</td>
<td>Content management</td>
</tr>
<tr>
<td>Create menu</td>
<td>Navigation management</td>
</tr>
<tr>
<td>Edit menu</td>
<td>Navigation management</td>
</tr>
<tr>
<td>Preview</td>
<td>Publishing</td>
</tr>
<tr>
<td>Publish</td>
<td>Publishing</td>
</tr>
</tbody>
</table>

Questionnaire 2 was given after the hands-on-session. Questionnaire 2 consisted of a set of questions to capture the participants’ perception of the assignment tasks they completed in the hands-on-session. The participants had to rate the difficulty of the tasks they completed for the assignment. The purpose of this questionnaire was to identify the end-users’ perception of the tasks they had completed in the hands-on-session. This questionnaire was designed to capture the participants’ reflection of their experiences with the CMS application and willingness to continue with EUD.
3.3.3 Participants

18 people voluntarily participated in the study. 12 participants out of 18 were male and 6 were female. 17 out of 18 were running a small business. One was in the government sector and was keen to participate in the workshop because she had a new responsibility to maintain a part of the council Web site. Most of the SME participants were doing the administrative work of their businesses by themselves. Some got help from their families. 10 out of 18 participants were over 60, 9 were in 40-60 age group and 5 were between ages 25-39. Only four of them had their own Web sites at the time.

3.3.4 Content Management System

Content Management System (CMS) supports development and maintenance of static Web sites. CMS is developed in CBEADS framework as an application. It uses the support functions provided within the framework such as authentication, user management, session management, and data base management.

The basic functionality provided by CMS includes create site, create page, edit page, manage meta-data, create menu, edit menu, upload image, preview page, preview site, and publish. These functions support the creation and maintenance aspects of a static Web site - presentation, content, navigation, and publishing. The description of the different tasks provided by the CMS tool that supports these categories of tasks is given in Table 3.2.
Table 3.2: Website development and Maintenance Tasks

<table>
<thead>
<tr>
<th>Task Category</th>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiation</td>
<td>Create site</td>
<td>In the initiation the folder structure and the database structure to store the Web site will be created.</td>
</tr>
<tr>
<td>Presentation</td>
<td>Select templates</td>
<td>Apply a selected template to the Web site.</td>
</tr>
<tr>
<td>management</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Content management</td>
<td>Add page</td>
<td>Add new Web pages to the Web site.</td>
</tr>
<tr>
<td></td>
<td>Edit page</td>
<td>Edit an existing page.</td>
</tr>
<tr>
<td></td>
<td>Upload images</td>
<td>Upload the images to be used in the Web site.</td>
</tr>
<tr>
<td>Navigation</td>
<td>Create menu</td>
<td>Create the menu of the Web site.</td>
</tr>
<tr>
<td>management</td>
<td>Edit menu</td>
<td>Change the menu of the Web site.</td>
</tr>
<tr>
<td>Publishing</td>
<td>Preview</td>
<td>Preview the Web site.</td>
</tr>
<tr>
<td></td>
<td>Publish</td>
<td>Upload (i.e. FTP) the Web site to Internet Service Provider (ISP).</td>
</tr>
</tbody>
</table>

The CMS application uses end-user development concepts: natural development, runtime programming and minimalism which were presented in chapter 2. This section explains the application of these concepts within the CMS application.

3.3.4.1 Natural development

The CMS application allows users to develop and maintain Web applications using application domain concepts rather than implementation domain concepts. The CMS application conceptualises the Web site as a collection of Web pages. Web pages are primarily accessed through a menu. A Web page consists of content. Content can be text, and image. Therefore the users have to deal with application domain concepts such as site, page, text, image, and menu.
The activities the end-users are required to carry out are similar to the activities the users have to perform in MS Word package. It is assumed that business end-users are capable of using office tools (such as word processing applications and excel) and e-mail. The findings on the capabilities of the end-users presented in section 3.3.5 prove that this assumption is valid. End-users can create and update the content of the Web pages using a What-You-See-Is-What-You-Get (WYSIWYG) editor shown in Figure 3.1. WYSIWYG text editor converts the text to the HTML code. Users can insert images by selecting the image from a list of available images. This is similar to browse and attach documents in e-mail applications.

Figure 3.1: WYSIWYG Editor
3.3.4.2 Runtime programming

The CMS application allows users to preview the Web page by clicking on the ‘preview’ option. Users can see the functionality of the site in one click. The switching between edit and run time mode is easy. Providing it as a ‘preview’ option, it clearly separates the two modes. Therefore, it avoids the confusion of two modes which is an issue with runtime programming as presented by Rode and Rosson (2003).

3.3.4.3 Minimalism

Minimalism refers to specifying only the minimum amount of information required to perform a task in a user interface. This was used as a guideline to design the instruction in user interfaces. A sample user interface is shown in Figure 3.2. The instructions are written in simple text using application domain concepts.

Figure 3.2: User Interface-Create Web site
3.3.5 Results

Objective 1: Benefits and motivations

![Bar chart showing percentage of participants by motivation factor: cost, ctrl, hobby.]

**Figure 3.3: Perception of benefits and motivations for EUD**

The question 4 in questionnaire 1 asks participants to list the reasons for their motivation in seeking the end-user development approach. The responses indicate that the motivations to EUD are reducing cost, control over application development (i.e. get the applications developed to their need instead adapting their business to match the applications) and as a hobby according to Figure 3.3, the strongest support (74%) was for reduction of cost. This can be related to the affordability of technology and IT professionals. It also shows that 50% would like to have the control over the application development to adapt it according to their needs of business.

Objective 2: Barriers

Question 5 in questionnaire 1 asks the participants to list the barriers to end-user development. The responses relating to barriers given in Figure 3.4 indicate that the biggest barrier is lack of know-how. Simply they don’t know a tool, where to start and how to start. Other barriers included time and risk associated with EUD. These results further indicated the need for processes and tools to be designed for efficient and effective use. Otherwise time will become the main hurdle due to the characteristics of
the SMEs. They carry out tasks such as administration, marketing and in extreme cases they work for their business in addition to managing their business.

![Barriers to EUD](image)

**Figure 3.4: Barriers to EUD**

**Objective 3: Capabilities of SMEs**

![Experience in IT](image)

**Figure 3.5: Experience in Computer Software applications**

According to Figure 3.5, all the participants were using application software such as word, excel and e-mail. Eight participants have completed a basic course on these applications. Seven participants had informal IT education. In other words they have
learnt to use computers and application software by themselves with support from different sources such as family, and friends. Three participants were using financial applications such as MYOB for accounting purposes. Only one had experience with development languages (VB, Java and Fotran).

![Experience in Web Development Chart]

**Figure 3.6: Experience in Web application development**

Figure 3.6 shows the experience in Web application development activities (design, development and maintenance) of participants. One participant has developed a Web site for his business using Dreamweaver and had experience with cascading style sheets. He has taught himself and developed a Web site for himself using a Web application tool freely available on the Web. He had an engineering education background and was helped by a family member. Two other participants had some experience using a CMS to upload their product catalogue or in general some maintenance work.

Table 3.3 shows the completion rates of assignment tasks according to the access log of the CMS application. 50% of participants completed all the tasks in the given assignment during the given time. Table 3.4 summarises the feedback given by the participants on the difficulty of the tasks in questionnaire 2.
Table 3.3: Task Completion rates

<table>
<thead>
<tr>
<th>Task</th>
<th>% of users complete the task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create site</td>
<td>66.6%</td>
</tr>
<tr>
<td>Add page</td>
<td>66.6%</td>
</tr>
<tr>
<td>Edit page</td>
<td>56%</td>
</tr>
<tr>
<td>Upload image</td>
<td>50%</td>
</tr>
<tr>
<td>Create/Edit Menu</td>
<td>60%</td>
</tr>
<tr>
<td>Preview</td>
<td>44.4%</td>
</tr>
<tr>
<td>Publish</td>
<td>50%</td>
</tr>
</tbody>
</table>

Table 3.4: Difficulty of tasks and rating description

<table>
<thead>
<tr>
<th>Task</th>
<th>Ranking</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td>Create site</td>
<td>22%</td>
</tr>
<tr>
<td>Add page</td>
<td>22%</td>
</tr>
<tr>
<td>Edit page</td>
<td>22%</td>
</tr>
<tr>
<td>Upload image</td>
<td>22%</td>
</tr>
<tr>
<td>Create/Edit Menu</td>
<td>22%</td>
</tr>
<tr>
<td>Preview</td>
<td>22%</td>
</tr>
<tr>
<td>Publish</td>
<td>22%</td>
</tr>
</tbody>
</table>

0 : Not answered.
1: I can't do it, it's too difficult.
2: I need some assistance the first time, but can do it on my own after.
3: I can do it after explanation and referring to the manual.
4: I can do it on my own by looking at the manual.

28% believed that they can create sites and edit pages by themselves referring to the manual. 33% said they were able to perform tasks such as add page, and upload image by themselves. 39% believe they can manage the menu and preview the pages while only
22% believe they could publish the site themselves. But after explaining the tasks, more than 50% believe they can complete all the tasks. This percentage was increased to 78% after a workshop. According to the feedback, it is identified that more than 50% participants were competent to work on content management, navigation management and publishing aspects of their Website after demonstration or by following the user manual.

**Objective 4: Feasibility of EUD**

From the participants’ responses to the question on continuation of a DIY approach in questionnaire 2, it is revealed that about 75% participants were willing to continue on end user development. However, of them 30% were not confident about continuing to end-user development. This can be related to their request for more training. These results together with the high rate of task completion establish the feasibility EUD among SMEs if they are provided with appropriate development tools.

### 3.4 Exploratory Study on End-user Development

This section presents a study on the benefits and issues of end-user participation during development of an Online Course Approval System (OCAS) (J. A. Ginige & A. Ginige, 2007). The study further reveals the practical relevance of end-user development.

#### 3.4.1 The project

OCAS is developed to support the course approval process at University of Western Sydney (UWS). The main objectives of the OCAS project were to: 1) Reduce redundant steps and optimise the process, 2) Reduce the overall time taken in processes, 3) Manage appropriate gate keeping and control steps that are needed for the administrative purposes of the university (J.A. Ginige & A. Ginige, 2007). The objective of the project
was to streamline the new course approval process, new unit approval process, course variation process and unit variation process. (In UWS terms a ‘unit’ is a ‘subject’ that is taught in a semester and a ‘course’ is collection of ‘units’.) The proposed system is also expected to be integrated with other information systems in the university such as the student management system named ‘Callista’. It was necessary to use the University wide user authentication mechanism that allows users to use their existing user name and password to access OCAS. OCAS is a complex Web information system. For example, in OCAS one business object has over 150 attributes; over 30 different users participate in the process flow in more than 40 different activities. Further, complex flow patterns such as parallel, splits, and joins were required in implementing the workflow. One challenge faced during the implementation of OCAS was the evolving nature of the project. These changes include information object, user interface, organisation structure and process (A. Ginige & J. A. Ginige, 2007). End-user development activities were used during the development of OCAS to accommodate these continuous changes.

3.4.2 End-user development activities

Different activities carried out by end-users during the development of OCAS are shown in Table 3.5. A number of sessions was organised between end-users and developers to enable the two groups to learn from one another. The mutual learning process supported developing a shared understanding of the problems the project aims to solve, while helping to anchor the proposed solution to the organisation (Bodker, Kensing, & Simonsen, 2004).
### Table 3.5: End-user development Activities in OCAS

<table>
<thead>
<tr>
<th>Activity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add user</td>
<td>Create the logins for new users by adding the required information such as name, e-mail address, login name and password.</td>
</tr>
<tr>
<td>Update user profile</td>
<td>Change the login information of existing users.</td>
</tr>
<tr>
<td>Add groups</td>
<td>Create a group to represent a role in the applications.</td>
</tr>
<tr>
<td>Assign users to groups</td>
<td>Assign existing users who are responsible for a role to the relevant group.</td>
</tr>
<tr>
<td>Assign functions to groups</td>
<td>Assign functions to groups which enable the users to access the functions that relate to their roles in the application.</td>
</tr>
<tr>
<td>Check and update state tables</td>
<td>Check the state table for the business logic and do minor updates to the business logic. For example specify who to be notified when a user submits a form in the system.</td>
</tr>
<tr>
<td>Write e-mail templates</td>
<td>Create the templates for e-mail messages to be sent when different actions happened in OCAS.</td>
</tr>
<tr>
<td>Populating object attributes</td>
<td>Insert the data to be used by OCAS. E.g.: List of existing courses</td>
</tr>
<tr>
<td>Defining project team templates</td>
<td>Specify the users in different roles related to each school within the University. This information was required to route the courses and units for each school.</td>
</tr>
</tbody>
</table>
3.4.3 Analysis of interview data

At the end of the project the end-user involved in these activities was interviewed. The objective of the interview was to identify the benefits and issues in end-user involvement in the project. Interview was semi-structured. The questionnaire used as the guideline for the interview is presented in Appendix 1. It allowed the end-user to explain her experience and issues in detail. Grounded theory was used to analyse the interview data. Calloway and Kanpp (1995) has identified that grounded theory is useful regardless of the granularity of analytical focus, the coding method, or the method of data generation. A coding technique was used to identify the key themes in the interview.

Key themes that emerged during the interview are benefits, and challenges. The benefits that end-user involvement brought to the project included fast adaptation of the system to the changing requirements, and reduction in development time. This also helped in the transition of the ownership of the project to the end-users. Personally, the interviewee felt that she had built up self-esteem from the exercise. She mentioned that the support available to her such as search function and runtime programming environment helped her to accomplish her tasks. One challenge she mentioned was that the developers did not listen to them in some instances which cost a lot later to the project. The other challenge was not having a tool to visualise the changes to the workflow. Therefore, it can be concluded that there should be an attitude change within the developers to get the maximum benefits of end-user involvement in the project. If the developers acknowledge the input they got from end-user participants in a project and appropriate tools were provided, EUD can reduce the development cost.
3.5 Requirements for an end-user development approach

The requirements for an end-user development approach suitable for business users, particularly SMEs were identified during the SME study and OCAS project in addition to the literature survey. The requirements include need of support for different types of Web applications required by SMEs, need to support specifications of Web applications at conceptual level, need for a common data repository to store the data used in different applications within the organisation, need to provide common login to all applications within the organisation, need to have a balance between DIY and professional developer that allows end-users to do the activities they are capable of while getting help from a professional developer to do the difficult tasks- a practical participatory development approach. These requirements are discussed in the following sections.

3.5.1 Support for different types of Web applications required by SMEs

In chapter 2, the eTransformation roadmap identifies the different types of Web applications and facilities required by SMEs: 1) Information centric, 2) Data intensive, 3) Workflow intensive, 4) Transactional.

In general, the different types of required Web applications can be broadly categorised into two groups: information centric, and process centric.

- Information centric: simple Web sites with unstructured information. The focus is on effective presentation of information.
• Process centric: Web sites that support business processes by enabling users to perform actions such as filling a form, approving a form, or carrying out a transaction. These can further be divided into two types;
  • Data intensive: The focus is on efficient presentation and management of structured data such as product catalogue.
  • Workflow intensive: The focus is on efficient automation of business process such as an order processing system.

3.5.2 Need to support the specification of Web applications at the conceptual level

In chapter 2, it was identified that the end-user developers’ mental model is closer to the application domain than the implementation domain. The study on capabilities of SMEs and the exploratory study on end-user development also confirmed the above finding. When analysing the capabilities of the SMEs it is found that they are competent in using Microsoft office software such as word, excel and e-mail. The completion rates of the activities imply that the end-user developers can accomplish the activities using WYSIWYG editor and graphical interfaces. The concepts end-users deal with in this study were Web site, Web page, menu based navigation and content. Further the end-user that participated in the OCAS case study mentioned that runtime programming is helpful to her since she is mostly goal driven to accomplish her tasks. When the end-users’ responses on tasks they accomplished were further analysed it was clear that they involve with conceptual level aspects such as templates, user interfaces, process and tasks. Therefore, it can be concluded that the inputs in an end-user development approach should be at conceptual level.
3.5.3 Need to support a common data repository

Business organisations need to maintain a common data repository. This will avoid duplication of data, a problem in stand alone applications. It is necessary to manage the integrity of data for efficient use of information. Further it is necessary to facilitate sharing the common data between different applications to achieve efficient use of information. This requirement was also identified by the stakeholders in the OCAS project. During the requirement gathering phase the stakeholders of the project complained about the difficulties arising from duplicate data in different systems within the University. Therefore, the end-user development approach should support different applications to use a common data repository. Creating and managing the data within the common repository need to be facilitated.

3.5.4 Need to support a common login to all applications within the organisation

As stated before, one of the requirements raised in OCAS was to reuse their existing logins. SMEs that worked with AeIMS also raised the need for common logins. This is an important usability requirement. Once a user logs in to a system he/she should be authenticated to all the functions which he / she is authorised to access. This helps to use end-users’ time efficiently.
3.5.5 Need to have a balance between DIY and professional developers for a practical participatory development approach

When CMS simplifies specifying aspects of Web applications at the conceptual level, it has to compromise between flexibility and simplicity. Based on the participants’ responses to the feedback questionnaire on enhancements to the CMS tool it was clear that the business users can’t be limited to a set of specific features to achieve simplicity. For, example, participants had stated their need for different templates rather than customising the available templates. Therefore, it is necessary to provide for a participatory development approach, so end-users are involved with activities they are capable of while not limiting the functionality of the application to their own capabilities. Professional developers should be involved in specifying the aspects if the users are not capable of doing the task to their expectation.

3.6 Conclusion

It is important to examine the practical relevance of the research question before conducting further research. This chapter presents two studies which give results encouraging enough to proceed to find answers to the research question. The chapter identified the following benefits, issues in end-user development, and capabilities of SMEs related to Web application development.

Benefits of EUD approach:

1. Reducing cost

2. Reduction in development time.
3. Control over application development (i.e. get the applications developed to their needs instead of adapting their business to match the applications)

4. Fast adaptation of the system to the changing requirements.

5. Self satisfaction.

Issues related to EUD:

1. Lack of know-how.

2. Limited time.

3. Neglecting the value of end-user participation by developers.

Capabilities of SME End-users

1. Can use application software such as word and excel.

2. Can specify the application using WYSIWYG editor and forms with some help.

3. Can conceptualise aspects such as content, navigation at the conceptual level.

The following set of requirements for an EUD approach suitable for business end-users was identified based on the issues and capabilities of SMEs:

1. Need to support different types of Web applications required by SMEs.

2. Need to support specifications of Web applications at conceptual level.

3. Need for a common data repository to store the data used in different applications within the organisation.

4. Need to provide common login to all applications within the organisation.

5. Need to strike a balance between DIY and professional developers for a practical participatory development approach.
The next two chapters present the two main constructs in this thesis: a conceptual model of Web applications and the meta-model based meta-design paradigm to support the requirements established in this chapter.
4 Conceptual Model of Different Types of Web Applications

4.1 Introduction

Chapter 3 established the requirements of an end-user development approach suitable for SME end-users. One of those requirements is to support specifying the Web application using aspects of the conceptual model of Web applications. Therefore, it is necessary to identify the aspects of different types of Web applications required by business end-users at the conceptual level. As identified in chapter 3, the Web applications required by SMEs can be broadly categorised into two groups: information centric, and process centric. Process centric Web applications mainly consist of data intensive and workflow intensive Web applications.

The existing literature on conceptual modelling of Web applications was reviewed to identify the aspects required to model different types of Web applications at the conceptual level. This review shows that the existing approaches to model Web applications mainly focused on a specific type of Web application such as data intensive or information centric. For example, the modelling approaches for Web developed in the early days such as HDM-Lite (Fratenali & Paolini, 1998), UWE (Koch & Kraus, 2002), W2000 (Baresi, Garzotto, & Paolini, 2001), OOHDM (Baresi, Garzotto, & Paolini, 2001) and WEBML 1 (Ceri, Fratenali, & Bongio, 2000) emphasized the aspects required to model data intensive Web applications such as data and hypertext. Later some of these approaches such as WebML, UWE and
OOHDM were extended to model process intensive Web applications. The modelling approaches have evolved over time to include business process modelling, customisation, and ubiquitous Web applications (Schauerhuber, 2007). At the time the author first reviewed the conceptual modelling approaches of Web applications, most of the approaches did not support workflow intensive Web applications. Further, the development environments supported by these modelling approaches focused on generating the code from the model as much as possible. As identified in section 2.6.3 these model driven development environments do not provide the necessary support for end-user development. These environments could be extended to support end-user development by embedding end-user development concepts (Valderas, Pelechano, & Pastor, 2007). The aspects in the conceptual model of Web applications have specified using diagrams or the alpha numeric format.

A set of specifications of Web applications written by end-users was examined. This study attempts to answer the third sub research question: “How do the end-users conceptualise Web applications?” In this study it is revealed that end-users naturally think in terms of some aspects of the conceptual model such as data and process. However, they do not specify aspects such as presentation. Therefore, a need to support creation of the applications is identified with some default values for aspects whenever possible.

This chapter presents a set of aspects that covers different types of Web applications. First the existing conceptual modelling approaches are reviewed. This review establishes the set of aspects used in different conceptual modelling approaches. Then example applications for each type of Web application are modelled to refine the set of aspects using WebML. The aspects are presented using diagram format where possible. After that an analysis of the usage of these aspects in a set of specifications for Web applications written by University
staff is discussed. Then the chapter concludes with a set of aspects and the requirements derived from the study of end-user specifications for an end-user development approach.

### 4.1.1 Conceptual model of Web applications

Many researchers have proposed different approaches to model the conceptual level of Web applications. These approaches are based on the E-R approach or the object oriented approach to model the data. Figure 4.1 shows the evolution of these different Web application modelling approaches. Different models use different terms to refer to the same aspects of Web application in the conceptual model. Most modelling approaches use UML to model different aspects. Some have used their own notations to model the aspects. Some of the approaches only focus on specific types of Web applications.

The literature survey of conceptual models of Web applications is focused on identifying a set of aspects to completely define the different types of Web applications. In this section, the different aspects covered in each conceptual modelling approach are summarised.
Figure 4.1: Web Modelling approaches (Schwinger & Koch, 2006)

HDM-Lite (Fratenali & Paolini, 1998) supports modelling of data-intensive Web applications using structure schema, navigation schema and presentation schema. HDM-Lite concentrates on efficient presentation of structured data. Structural schemas are used to model the structural properties of the basic objects that make up the application. Navigation schemas model the actions available to move from one object to another one (traversal schema) and the access paths to reach the objects of the application (access schema). The presentation model specifies the way application objects are presented to the user.

In OOHDM (Schwabe, Rossi, & Barbosa, 1996), the conceptual model is derived based on Object Oriented Modelling (OMT) principles. The data model is defined using UML class diagrams and relations. The navigation model is also modelled as two classes, the navigation class schema and the navigation context schema. The navigational context indicates possible
navigation sequences to help users complete a task. The navigational class schema represents a view of the data in the form of a proprietary text-based query language. The abstract interface design is focused on supporting the user to activate the functionality and to navigate through the application. It represents abstract interfaces for navigational objects such as nodes, links, and access structures. The presentation model specifies the look and feel of the application, including layout, font, colour and graphical appearance. Later, an ontology based approach has been used to specify the interfaces (Rossi & Schwabe, 2006).

WSDM (Paterno, 2000) approach is based on the definition of a new task model on the ConcurTaskTree notation (Paterno, 2000) and refines it to specifically deal with Web application concerns. Recently, WSDM is evolving towards the semantic Web (Casteleyn, Plessers, & De Troyer, 2006). A task is decomposed into elementary tasks arranged in a temporal order. An object chunk is created for each elementary task, which (formally) models the necessary information and functionality needed to complete the task. Moreover, an object chunk can have associated object chunk functions, which allow modelling of the system functionality (e.g., instance creation and select functions). A set of task navigation models is defined based on the task models and the object chunks to specify how the user will be able to perform the tasks in the Web site. At presentation level, the site structure model maps the concepts modelled at hypertext level onto pages. In addition, WSDM proposes page models defined for each separate page in the site structure model, which allows for positioning of page elements. WSDM supports modelling data-intensive Web applications.

WebML1 (Ceri, Fratenali, & Bongio, 2000), the original version of WebML, supports modelling of data-intensive Web applications. This approach consists of models such as a
structural model (i.e. data model), a hypertext model (composition and navigation models),
presentation models, personalized models and operation models. The WebML data model is
used to specify the domain objects and relationships between domain objects. The hypertext
model in WebML1 defines the pages and their internal organization using content units that
is displayed on the pages, links between pages and content units. The operational model is
defined for content management and login logout procedures (Brambilla, Comai, Fratenali,
& Matera, 2008). Later, WebML integrated with BPML to model workflows in Web
applications (Brambilla, Ceri, Comai, & Tziviskou, 2005). The data model is extended to
describe the meta-data necessary to track the execution of the business process, both for
logging and for constraint evaluation. The hypertext model is extended to specify the
business activity boundaries and the workflow dependant navigation (Brambilla, Comai,
Fratenali, & Matera, 2008).

W2000 (Baresi, Colazzo, Mainetti, & Morasca, 2006; Baresi, Garzotto, & Paolini, 2000) is a
modelling approach used to model data-intensive Web applications. W2000 includes the
information (data) model, navigation model and operation model. The data model specifies
the data entities and relationship among them. The access diagram is used to specify who
can access what information. The navigation model in the W2000 model controls how the
users can navigate through the content based on the relationships in the information model.
The operation model specifies the functions that users perform on the data objects. The
presentation model specifies the information elements on a page such as forms, navigation
elements and labels (Baresi, Colazzo, Mainetti, & Morasca, 2006).

The Hera approach (Houben, Barna, Frasincar, & Vdovjak, 2003) supports modelling data
intensive Web applications. It consists of models for the content (i.e. data), hypertext, and
presentation levels. The domain model (i.e. data model) was first based on the ER-model. Later it evolved to Resource Description Framework (Schema) - RDF(S) (RDF Primer (XML) 1.1 2004) thus supporting the engineering of semantic Web information systems. It provides a proprietary graphical notation for modelling the data (Frasincar, Houben, & Barna, 2006). The domain model of Hera is based on concepts, attributes concept relationships and media types. The hypertext level (i.e., the application model) and the presentation are mainly based on slices and slice relationships, which can be either compositional or navigational. The presentation level is based on a hierarchy of regions allowing for positioning of slices from the hypertext level for their presentation. A region represents a rectangular part of the display area and has associated with it a layout manager for positioning elements and a specific style. The recent introduction of concepts for modelling workflow-based Web applications (e.g., a task model and a workflow model) (Barna, Frasincar, & Houben) allow for modelling processes.

The UML-based Web Engineering (UWE) (Koch & Kraus, 2002) approach is based on RMM, OOHDM, and WSDM. For example, it uses some graphical elements from RMM such as index and guided tour. It separates the construction of conceptual, navigation, and presentation models that stems from OOHDM, and continues with the user-centred approach of WSDM. The UWE conceptual model (a.k.a. data model) is composed of user models and domain models. The UWE user model includes the structures directly related to the individual users of the application. The domain model consists of domain objects related to the applications (Koch, Kraus, Cachero, & Melia, 2004). The UWE process model is used to model the user actions. The user actions are captured using the use cases in UWE (Koch & Kraus, 2002). Use cases are explained using behavioural diagrams or textual format. The process model consists of structural view, behavioural view and navigation view. The
navigation model is used to indicate browsing possibilities with in the applications based on the use cases. Navigation model also consists of a set of access structures required for the navigation. Process model is used to model the flow of activities within a use case. The structural model is derived based on the UWE data model to capture process related information. The behaviour view is used to model the flow of activities within the use cases. The presentation model of UWE is used for the specification of the logical presentation of the Web application. The presentation model is further refined to specify the elements of physical layout such as font and colour. The presentation model consists of structural view and user interface view. The structural view is focused on modelling the structure of presentation space - dividing and grouping the presentation elements to be displayed in the same page. The UI view is focused on the details of user interface elements presented in a page.

The OO-H method (Gomez & Cachero, 2003) supports modelling process centric Web applications, specifically data intensive applications. The approach comprises different models for the data, hypertext and presentation. Its data model consists of two packages called user model and domain model. The user package includes structures directly related to the individual user, such as address. The domain model includes information related to domain objects such as products and orders (Koch, Kraus, Cachero, & Melia, 2004). UML class diagrams are used to model these packages. In the OO-H approach, the use-case diagram and the storyboard are used to decompose the processes of the system into tasks. The process model defines the inner flow of control of non trivial use cases. It provides a way to partition and characterise groups of operations. UML activity diagrams are used to model processes. The default navigation model can be mapped from the activity diagram. The navigation model in OO-H is defined using a Navigation Access Diagram (NAD). NAD
is developed for each user type and consists of navigation paths and the operations the user can activate. It consists of navigation classes which represent the views on classes of the domain model. The presentation model specified using an Abstract Presentation Diagram (APD) and interpreted as the sitemap of the Web application consisting of a set of “abstract pages”.

The OOWS (Pastor, Fons, Pelechano, & Abrahao, 2006) method is an Object Oriented (OO) approach to model data intensive Web applications. OOWS is built on the OO-method (Pastor, Gomez, Insfran, & Pelechano, 2001) that supports modelling information systems. The OO-method consists of models to specify the structure and functions of dynamic applications. The structural model (data model of OO-method) defines the system structure (its classes, operations and attributes) and relationships between them. The dynamic model specifies the object-life sequence and the object interactions. The functional model captures the operations on the objects that define the services provided in the system. OOWS extends the OO-method to model Web information systems. The additional models include the user model, the navigation model and the presentation model. The OOWS user model defines the different kinds of users that interact with the systems and what they can access in the system. The navigation model is defined over the structural model to specify the data and operations users can access. The presentation model is defined based on the navigational model and it uses system–user interaction units to define the presentation properties.

Table 4.1 summarises the concepts covered by the different conceptual modelling approaches. Most of the existing Web modelling approaches neglect business processes at the early stage of modelling (Jakob, Schwarz, Kaiser, & Mitschang, 2006; Schmid & Rossi,
2004). Such approaches have embedded aspects such as the process model and the user model within other aspects rather than defining these separately during the analysis phase of Web applications. Also it is possible to abstract a composition model at a higher level than the granular elements such as label, text, list box for process intensive Web applications (Liang & A. Ginige, 2006a). The other important finding is that all the modelling approaches are still evolving to cover different types of Web applications. Therefore, it is required to identify the attributes of the aspects of Web applications at the conceptual level.

WebML is identified as a near complete approach since it evolved to support most of the aspects required to model different types of Web applications (Brambilla, Comai, Fratenali, & Matera, 2008). The WebML approach to model Web applications at the conceptual level is explained in section 4.1.2. WebML is used to model the example applications. These examples help to verify the expressive power of the aspects to specify the Web applications completely. WebML notations have been modified or other notations have been used when WebML is not adequate to express the aspects.
Table 4.1: Summary of Web application modelling approaches

<table>
<thead>
<tr>
<th>Modelling Approach</th>
<th>Composition</th>
<th>Presentation</th>
<th>Navigation</th>
<th>Data</th>
<th>Task</th>
<th>Workflow</th>
<th>Access Control</th>
<th>Personalisation</th>
</tr>
</thead>
<tbody>
<tr>
<td>WebML</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>HDM-Lite</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OOHDM</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>OOWS</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OO-H</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UWE</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W2000</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hera</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WSDM</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4.1.2 WebML modelling approach

4.1.2.1 WebML data model

WebML uses the entity-relationship model or UML class diagram to model the data. The fundamental elements of the WebML data model are entities which are the containers of data elements and relationships which are the semantic connections between entities (Brambilla, Comai, Fratenali, & Matera, 2008). The named properties of entities are called attributes. Attributes have associated type. Entities can be organized into generalization hierarchies. The relationships can be restricted by means of cardinality constraints.
4.1.2.2 WebML hypertext model

The WebML hypertext model defines the front-end interfaces of Web applications, which are shown to a user in the browser. The interfaces are specified using the pages. The internal organisation of the pages is defined in terms of components (called content units) for displaying content. WebML1 defines two sub component types called composition model and navigation model (Ceri, Fratenali, & Bongio, 2000). The composition model specifies the content units within pages. WebML has five predefined content units called data unit, multi data unit, index unit, scroller unit and entry unit which are shown in Figure 4.2 (Brambilla, Comai, Fratenali, & Matera, 2008). Data units represent one or more instances of entities of the data model, typically selected by means of queries over the entity attributes or over relationships. Multi data units represent some of the attributes of a set of entity instances. Index units present a list of descriptive keys of a set of entity instances which enable the selection of one of them. Scroller units enable the browsing of an ordered set of objects. Entry unit allows publishing a form to collect input values from the user.

<table>
<thead>
<tr>
<th>Data Unit</th>
<th>Multidata Unit</th>
<th>Index Unit</th>
<th>Scroller Unit</th>
<th>Entry Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="data-unit.png" alt="Data Unit" /></td>
<td><img src="multidata-unit.png" alt="Multidata Unit" /></td>
<td><img src="index-unit.png" alt="Index Unit" /></td>
<td><img src="scroller-unit.png" alt="Scroller Unit" /></td>
<td><img src="entry-unit.png" alt="Entry Unit" /></td>
</tr>
</tbody>
</table>

Figure 4.2: WebML Content units (Brambilla, Comai, Fratenali, & Matera, 2008)

The WebML navigation model supports the definition of links between pages and content units that support information location and browsing. Links between units are called
contextual, because they carry some information from the source unit to the destination unit. On the other hand the links between pages are called noncontextual.

WebML2 defines operation units in addition to the composition and navigation units defined in WebML1. These operation units are used to specify operations required for content management (create content, delete content, modify content, connect—add relationships and disconnect—drop relationships) and user’s login/logout procedures (Brambilla, Comai, Fratenali, & Matera, 2008). WebML operations are shown in Figure 4.3.

<table>
<thead>
<tr>
<th>Create Unit</th>
<th>Modify Unit</th>
<th>Delete Unit</th>
<th>Connect Unit</th>
<th>Disconnect Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Create Unit Icon" /></td>
<td><img src="image2" alt="Modify Unit Icon" /></td>
<td><img src="image3" alt="Delete Unit Icon" /></td>
<td><img src="image4" alt="Connect Unit Icon" /></td>
<td><img src="image5" alt="Disconnect Unit Icon" /></td>
</tr>
</tbody>
</table>

**Figure 4.3: WebML operation units (Brambilla, Comai, Fratenali, & Matera, 2008)**

### 4.1.2.3 WebML workflow model

WebML 4 (Brambilla, Ceri, Fraternali, & Manolescu, 2006) is integrated with BPML ("Business Process Management Language", 2006) to specify the workflow. The main concepts of WebML are process, actor, activities, constraint, sequence and split/join. The notations used for main constructs are given in Figure 4.4
Figure 4.4: BPMN main construct (Brambilla, Ceri, Fraternali, & Manolescu, 2006)

- Processes: high-level descriptions of the work to be globally performed.
- Actors: the users performing the work.
- Activities: the units of works comprising a process, typically performed by a single actor.
- Constraints: the logical precedence among activities. BPMN constraints assume a variety of forms:
  - Sequence: a sequence is a combination of two (or more) activities that can be executed only in sequential order (i.e., one after another). One activity must finish before the next one can start.
  - AND-split / AND-join: at the split point, the execution flow is split in two (or more) parallel branches, thus enabling mandatory parallel execution of two (or more) activities. All the branches must be executed. Parallel execution is not strictly enforced.
4.1.2.4 WebML access control model

WebML user schema clusters specify the data about users and their access rights to data. Users belong to groups. The SiteView allows users (and user groups) to be associated with specific hypertexts (Brambilla, Comai, Fratenali, & Matera, 2008). An example site view which shows the functions that the applicant can access are shown in Figure 4.5.

![Diagram of a site view in WebML](image)

Figure 4.5: An example Site View which gives the access control aspect (Brambilla, Ceri, Fratenali, & Manolescu, 2007)

4.1.2.5 WebML personalisation model

The WebML personalisation model consists of entities from the data model associated with the user entity by means of relationships expressing user preferences for some entity instances, or the user’s ownership of some entity instances (Brambilla, Comai, Fratenali, & Matera, 2008). The next section presents the aspects required to model different types of Web applications.
4.2 Aspects required to model information centric Web applications

The objective of an information-centric Web application is to publish information effectively. Major aspects of an information-centric Web application can be modelled using the presentation model, the navigation model, the access control model (optional), and the personalisation model (optional).

The Actif Toolcraft Web site which can be found at http://www.actiftoolcraft.com (Home page of the site is shown in Figure 4.6) is modelled to check the completeness of the aspects. The Web site provides the information on their company, products, services and machinery.
4.2.1 Presentation model

The presentation model defines the look and feel of a Web application. It dictates the layout and graphic appearance of all generic elements that appear on a page. The presentation model is defined at two levels: site level and page level. The template and style settings of the presentation model at site level apply to all the Web pages of the site. The template and style settings for a custom page are defined at page level. The table format is used to specify the presentation properties and a diagram is used to specify the page lay out.

Figure 4.7 shows the site level template for the Actif site. The Actif template consists of a left side menu for navigation, a company banner as the header with the company logo at the left, the footer at the bottom. Content had a page heading and one or two images appeared at the right hand side column within the content section.

Table 4.2 gives the site level presentation properties of the Actif site. Different font size, font family, font style, and colour were used for the page heading, link and text at the site level.

Figure 4.7: Site Level Template of Actif
Table 4.2: Site level Presentation properties

<table>
<thead>
<tr>
<th>Element</th>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body text</td>
<td>Font-family</td>
<td>Arial</td>
</tr>
<tr>
<td></td>
<td>Font-size</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>color</td>
<td>Black</td>
</tr>
<tr>
<td>Header 1</td>
<td>Font-family</td>
<td>Verdana</td>
</tr>
<tr>
<td></td>
<td>color</td>
<td>Blue</td>
</tr>
<tr>
<td></td>
<td>Font-size</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>Font-style</td>
<td>Bold</td>
</tr>
<tr>
<td>Link</td>
<td>color</td>
<td>Blue</td>
</tr>
<tr>
<td></td>
<td>Size</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>Font-style</td>
<td>Bold</td>
</tr>
<tr>
<td></td>
<td>Font-family</td>
<td>Verdana</td>
</tr>
<tr>
<td>Visited Link</td>
<td>color</td>
<td>Purple</td>
</tr>
<tr>
<td></td>
<td>size</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>font-style</td>
<td>Bold</td>
</tr>
</tbody>
</table>

4.2.2 Navigation model

Two types of navigation exist in information centric sites: primary navigation and secondary navigation. Primary navigation is the main method of accessing information in various pages or sections of pages in the Web site. Typically the menus are used as means of primary navigation. Another possible way of primary navigation is using an unstructured or structured search mechanism (A. Ginige, 2008). Within pages, content can link to access related information on the same site or external sources. These links are defined as secondary navigations. Another example of specifying a secondary navigation link is to have a glossary and a rule which states that if the user on any page clicks on a word described in the glossary, this description should appear in a pop-up window (Ceri, Fratenali, & Bongio, 2000).
Navigation can be specified using the WebML notations. The attributes of the navigation aspect are source page and destination page.

The primary navigation for an actif site is given in Figure 4.8. In this simple site product, machinery, services, links, and about us pages are accessed through the main menu.

![Diagram of primary navigation](image)

**Figure 4.8: Primary navigation of Actif site**

### 4.2.3 Composition model

Pages of an information centric Web application consist of a combination of text, video, graphic, images, sounds and secondary navigation links. The composition models of information centric Web applications are unstructured and generally specified at page level. The attributes of the composition model are area and content. The attributes can be specified using table format.
Table 4.3 gives the composition model of the Actif home page which consists of text and an image. The specific areas are introduced in the presentation model of the site (Refer to Figure 4.7).

**Table 4.3: Composition Model of Actif site**

<table>
<thead>
<tr>
<th>Area</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content</td>
<td><strong>Actif Toolcraft</strong></td>
</tr>
<tr>
<td>Area</td>
<td>Actif Toolcraft Pty Ltd is a well known toolmaking company located in Western Sydney NSW. The business has been in operation since 1987 and their focus is to design and manufacture high quality plastic moulds, blow moulds, precision engineering and high/ low volume production work. Actif Toolcraft has actively invested in modern technology, which has allowed for increased efficiency and the production of quality precision engineering components and tooling. By being fully equipped with modern in-house technology, Actif Toolcraft has been able to combine fine toolmaking skills with the latest computerized machinery. To view further information about Actif Toolcraft's products, services or machinery, simply follow the links provided on the left navigation bar. If you wish to contact Actif Toolcraft for additional enquiries, all contact information can be found via the &quot;Contact Us&quot; link.</td>
</tr>
</tbody>
</table>
4.2.4 Access Control model

In an information centric application the access control model specifies the relationships between users/group of users and pages, such as what information users can view. Authorised users can access these pages. WebML has modelling notations relating to access control model in process centric Web applications –“ who can access what functions”. This is adapted to show who can access what pages in the information centric Web applications. The attributes of the access control aspect are user/user group and page. This can be specified using the WebML site view.

For example if the Actif site can be extended with specific information only for its collaborative partners then the access control model for the site can be specified using a “view“. In WebML, a view specified what can be accessed by particular user group / user. In the “Collaboration group” site view it shows the pages that a group can access as given in Figure 4.9. In this case, the users of the collaboration group can access the “Current projects” page.
4.2.5 Personalisation

An information centric Web site may customise the composition model, navigation model or the presentation model based on attributes of the user/user group. The attributes of the user or user groups can be shopping suggestions, list of favourites, and presentation properties (Ceri, Fratenali, & Bongio, 2000). The attributes are specified based on the context of the user. For example, context can be the location of the user or the navigation history of the user. Personalized composition models, navigation models or presentation models per user or user group can be derived from these attributes.

4.3 Aspects required to model data intensive Web applications

The objective of data intensive Web applications is to manage information effectively. Therefore these are additional models in the information centric sites: data model, and action model (optional). The composition model of data intensive Web applications is different to information centric ones. This section discusses the changes or additions to the models presented in section 4.2.
The product catalogue, in the Bosco site (http://www.boscooffice.com.au/) is discussed as an example of a data intensive Web application which provides information about the products of the company. The company produces products belonging to different product groups such as doors, cabinets, and shelving. Company has different products in these groups. For example, it has products called cupboards, lockers and book cases which belong to the product group cabinet. Each product comes with different dimensions which are called variations or sub products. An example product page and sub product page from the Bosco site are shown in Figure 4.10: A product page from Bosco siteFigure 4.10 and Figure 4.11.

Figure 4.10: A product page from Bosco site
Figure 4.11: A sub product page from Bosco site

Figure 4.12 shows the simple structure schema for the product catalogue. Products Groups are composed of products. Each product can have one or more sub products. To publish this information on the Web, it is necessary to specify composition and navigation models (Ceri, Fratenali, & Bongio, 2000).

![Structure Schema](image)

Figure 4.12: Structure Schema used in Bosco Catalogue

### 4.3.1 Data model

A data model defines relevant objects and relationships among objects used in a Web application. A data model is the key concept in data-intensive Web applications. The attributes of data aspects are objects and relationships. Objects can be described using their
attributes. WebML has defined attribute types such as string, float, integer and image (Ceri, Fratenali, & Bongio, 2000). The list of predefined types can be extended with more high level attributes such as description, photo and e-mail as specified in Smart Business Object Modelling Language (SBOML) (Liang & A. Ginige, 2006b). SBOML is detailed in Appendix B. This high level abstract of attribute types will help to improve naturalness in modelling the data aspect. Therefore, in modelling the data, an object model is used with SBOML data types. Relationships can be explained in terms of objects which are related. Object relations have constraints called multiplicity which specify how many instances of an object can be related to one instance of another object.

The data model for a product catalogue is shown in Figure 4.13. The product group has many products and a product has many sub products. Attributes for three objects, product group, product and sub product are specified using SBOML data types.

![Data model diagram](image)

**Figure 4.13: Data model**

### 4.3.2 Navigation model

The primary navigation in a data intensive Web application is usually specified based on the data model. The navigation aspect of data intensive Web applications can be specified using data objects or database query which specify what is to be shown in each page. The attributes
of the navigation model are source page [data object/query] and destination page [data object/query]. This can be modelled using WebML notations as shown in Figure 4.14.

In the Bosco site the product catalogue consists of two menus: the first one is a product menu and the second one is a sub product menu. If a product is selected in the product menu, the detail of the product is displayed with a relevant “sub product” menu. Once the sub product is selected in the “sub product” menu sub product data are presented. The navigation model of the Bosco product catalogue is defined based on the data model given in Figure 4.13.

![Figure 4.14: Navigation Model of Product Catalogue](image)

### 4.3.3 Composition Model

In a data-intensive Web application, the composition model specifies the information elements that may appear in the Web site and the pages, i.e., containers by means of which information is actually clustered for delivery to the user (Ceri, Fratenali, & Bongio, 2000). These components are referred to as UI components. Data-intensive Web applications may consist of components such as forms, tables, menus and charts (Liang & A. Ginige, 2007).
UI components consist of UI guide, UIElementGroup, and or UIElements, and UI Actions. UI elements can be WebML data units, which store or present data. UI is associated with the business object which binds with the interface. UI elements can be in input mode or output mode. Form consists of UI elements in input mode, i.e. called entry units in WebML. A table consists of UI elements in output mode. It can be WebML index, single data, multi data, or search data unit. UIGuide provides the guidelines for using the particular interface. For example, in a form interface, the guidelines can help the users to understand the purpose of the form. In a table interface, the guidelines can help users to interpret the report properly. A UI element in input mode can also be associated with a help tip. Help tips help users to enter the values of the form UI element correctly. The WebML entry unit can be extended by adding help information for each field of entry unit. The composition model specifies the UI components associated with different areas of a page.

The composition model of the sub product pages in the Bosco site is given in Figure 4.15. The composition model consists of product description, list of features (image and feature name), secondary navigation link to download the product specification from the site, a dimension chart of the products, and an image of the product.
4.3.4 Action model

Jacob et al. (Jakob, Schwarz, Kaiser, & Mitschang, 2006) define the minimum actions a data-intensive Web application should support in order to add new content objects or new relations between content objects, alter existing content objects or existing relations between content objects, delete existing content objects or existing relations between content objects, filter content objects according to conditions and sort content objects by specified criteria. Then there can also be some other actions such as send notification associated with data-intensive Web applications. Normally, a composition model for each action has to be specified. The attributes of the action model are the actions and data objects related to the actions.
The action model of a product catalogue consists of actions: manage (add/edit/delete) product group, manage (add/edit/delete) product, and manage (add/edit/delete) sub product. Figure 4.16 shows the action model for manage products.

![Diagram of action model of product catalogue](image)

**Figure 4.16: Action model of product catalogue**

In add action, the product information needs to be entered and then the product should be associated with a product group. Once the data storing is completed the user should be given the success message. If a specific product needs to be updated, then the product has to be selected and then a form will appear with existing information. The user can change the information and click on update action. Related data objects will be updated and the user should be given the success message. In delete action, once the product needs to be selected, the product data from the database are flagged for deletion. Since the end-users are maintaining the Web applications, this will prevent deletion of data by mistake. The composition models for the entry unit and response unit need to be specified as given in 4.3.3.
4.3.5 Access control model

In a data-intensive Web application, the access control model specifies which pages or the actions a user/user group can access. The attributes of an access control model are user/user group and actions.

The access control model for administrators of a product catalogue is shown in Figure 4.17. Admin has access to three actions manage product, manage product group and manage sub product. The WebML site views are used to model the access control. However, the specific notation “A” is used to denote the actions.

![Admin Site View](image)

**Figure 4.17: Access Control Model of Product Catalogue**

4.4 Aspects required to model workflow intensive Web applications

The objective of workflow intensive Web application is to automate the business processes effectively. According to the Workflow Management Coalition–WfMC (Feb 1999, p8),
“Workflow is an automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules”.

Therefore, a workflow model is required to specify workflow intensive Web applications, in addition to the models required for data-intensive Web applications. Further the access control model has to be extended to support workflows. This section discusses the changes or additions to the models presented in 4.2 and 4.3.

The Online Course Approval System (OCAS) which can be found at http://ocastest.uws.edu.au is modelled to check the completeness of aspects related to workflow intensive Web application. OCAS is developed to support the course approval process in University of Western Sydney.

4.4.1 Workflow model

In this thesis workflow is conceptualised as a sequence of actions. This is called activity based workflow modelling ("OpenFlow"). Jablonski and Bussler (1996) identify five elements of a workflow specification: functional, behavioural, informational, organisational, and operational elements. The functional aspect describes what should be done in the workflow: it specifies the actions in the workflow. The behavioural aspect describes when something should have happened: it gives the execution order and dependencies (control flow) of actions in the workflow. The informational aspect describes the data used in the workflow. The organisational aspect describes who should do the work in the workflow. The operational aspect describes how the workflow management system should interact with its environment. It describes the methods for accessing or invoking external applications (e.g. interaction mode, invocation mode, parameters) and how to communicate with human users. These five workflow elements can be specified using activity, event, gateways, message flow,
association, data and swim lanes in BPMN notation. The attributes of the workflow model are user, state, action, next state, pre conditions, do activities.

Notification of Intention (NI) is one of two sub processes of the course approval process in OCAS. Workflow diagram for NI process in OCAS is shown in Figure 4.18. The NI process starts with an academic user submitted NI which briefly outlines the new course he/she intends to propose. NI should be approved at the school level, and college level and university level. NI should finally be approved by the Vice Chancellors Advisory Committee (VCAC) before starting a full course proposal.

Any user who belongs to an academic group can submit a NI for a new course. This is a state independent action which is available all the time. Once NI is submitted, it'll be sent to the Head of School (HOS) of the school the academic belongs to. HOS can approve, reject or send back NI for modifications. HOS approval is a state dependant task. All state dependant tasks which require the user’s action in OCAS appear on a separate page called ‘my task’. If it is approved it will be sent to Dean of College where the school belongs to. Once NI is approved by the Dean then it is sent to Pro Vice Chancellor Academic (PVC academic) and VCAC Secretary. Either one can approve NI.

The university has many colleges and colleges have many schools. Therefore instance level access control needs to be modelled to identify the specific roles within the college or the school to be associated with instance of data. For example, when an academic of the School of Computer Science and Mathematics submits a NI, it should be sent to the HOS of that school. However, the organization roles and structure can change dynamically (A. Ginige & J. A. Ginige, 2007). Therefore, at the conceptual level only the organisation structure is modelled.
4.5 Study on end-user specifications

Thirty two proposals submitted by staff members in the university to develop or modify existing Web applications were analysed. These staff members belong to different departments of the University and none of them is a Web developer. From these 32 proposals 17 were for new Web applications or enhancements to existing Web applications. The rest was for new hardware or hardware upgrades. The proposals for Web application were further analysed to identify the concepts they have used. Twelve out of seventeen users specified objects and some attributes for the objects. Since in most cases users have some sort of manual system, they had an understanding of data to be managed in the proposed information system. But no one specified navigation, personalisation, or presentation models. Only one specified the composition model. 50% specified processes. 80% specified
user model with access control and some attributes for user groups. Table 4.4 shows usage of aspects as a percentage out of 17.

**Table 4.4: Usage of aspects in end-user specifications**

<table>
<thead>
<tr>
<th>Aspect</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>70%</td>
</tr>
<tr>
<td>Navigation</td>
<td>4%</td>
</tr>
<tr>
<td>Composition</td>
<td>5%</td>
</tr>
<tr>
<td>Presentation</td>
<td>0%</td>
</tr>
<tr>
<td>Process (Workflow/ Action)</td>
<td>50%</td>
</tr>
<tr>
<td>Access Control</td>
<td>80%</td>
</tr>
</tbody>
</table>

This is only a sample of end-users. However, this sample of end-users confirms Rode’s conclusion on the end-user mental model. They want systems to store data and then manipulate data. They are not concerned about critical aspects such as presentation and hypertext. All of that will come as usability issues of the system when they use the systems. However, when they become the owners of the system they need to be able to create or modify the system. Then they also need to consider other aspects such as the hypertext model. When designing tools for end-users to develop Web applications it is required to incorporate the default values for aspects such as presentation model and navigation model. This will help to match the tools to the end-user mental model.

**4.6 Conclusion**

Analysis of existing approaches to conceptual modelling helps to identify a set of aspects required to model different types of Web applications required by SMEs. In this analysis it was clear that WebML has evolved to support many aspects required to model these
different types of Web applications. The modifications to the WebML models presented in this chapter help to express the conceptual model of Web applications completely. Modifications to WebML models are detailed in Table 4.5.

Table 4.5: Summary of Modifications to WebML

<table>
<thead>
<tr>
<th>Model</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Presentation</td>
<td>Presentation model specified in section 4.2.1 is added to WebML models.</td>
</tr>
<tr>
<td>Data</td>
<td>SBOML data types are added to WebML data types. SBOML is detailed in Appendix B.</td>
</tr>
<tr>
<td>Composition</td>
<td>Composition model for information centric Web applications specified in section 4.2.3 is added. Composition model for data intensive Web applications is modified to include different UI component types as explained in section 4.3.3.</td>
</tr>
<tr>
<td>Access Control</td>
<td>Action notation is used to specify the access control aspect related to data intensive and workflow intensive Web applications as given in section 4.3.5.</td>
</tr>
</tbody>
</table>

The complete set of models and the attributes required to specify the different aspects required to model Web applications at the conceptual level are given in Table 4.6.

Table 4.6: Models and Attributes of different aspects

<table>
<thead>
<tr>
<th>Model</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Presentation</td>
<td>Area layout, presentation property</td>
</tr>
<tr>
<td>Data</td>
<td>Object, relationships</td>
</tr>
<tr>
<td>Access control</td>
<td>User group, function/page</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>Workflow</td>
<td>State, action, activity, entry condition, exit condition</td>
</tr>
<tr>
<td>Navigation</td>
<td>Source (page/ object), destination (page/object)</td>
</tr>
<tr>
<td>Composition</td>
<td>Area, content/ UI component</td>
</tr>
<tr>
<td>Action</td>
<td>Entry unit, operation units, response units</td>
</tr>
<tr>
<td>Personalisation</td>
<td>User/ user group, condition</td>
</tr>
</tbody>
</table>

The meta-model based meta-design paradigm presented in chapter 5 uses different end-user development technologies and tools to specify the attributes of these models.

The author has examined application specifications written by staff members in different departments in the university. In this study it is revealed that end-users naturally think in terms of some aspects such as data and process. The need for modelling other aspects will arise later during the use time of the application. Therefore, an end-user development application should allow end-users to specify the applications with minimum aspects. The next chapter explains the meta-model based meta-design paradigm which accommodates this requirement and the other requirements specified in chapter 3.
5 Meta-design Paradigm and Meta-model

5.1 Introduction

Chapters 2, 3 and 4 identify the requirements of business users, particularly SMEs, for a suitable end-user development approach to develop and maintain Web applications. These requirements are: 1) Support for different types of Web applications required by SMEs; 2) Need to support the specification of Web applications at the conceptual level; 3) Need for a common data repository to store the data used in different applications within the organisation; 4) Common login to all applications within the organisation; 5) Balance between DIY and professional developer that allow end-users to perform the activities they are capable of while getting help from a professional developer to do the difficult tasks; 6) Support to create Web applications with minimum aspects. Chapter 4 presents aspects of Web application at the conceptual level which can be used as input to the Web development framework to provide a natural development approach.

Developing a meta-design paradigm with an appropriate technical infrastructure, a set of meta-design tools, and a socio technical environment can support the requirements of an end-user development approach to build Web applications. As explained in section 2.3.2.4, the meta-design paradigm should provide the objectives, techniques, and processes for
creating new media and environments allowing ‘owners of problems’ (that is, end-users) to act as designers. This thesis proposes creating an application based on the meta-model as a suitable approach for a meta-design paradigm for end-user developers to use to develop Web applications.

The meta-model of Web applications is used to create an ‘open system’ which can be designed and modified by end-user developers according to their evolving requirements after deployment of the system. Meta-models of Web applications are created based on the patterns of different types of Web applications. A meta-model can be used to generate the Web applications end-users require by specifying the attributes of the aspects at the conceptual level. For example, the meta-model of workflow intensive Web applications can be instantiated to create a leave application system where the employees can apply for leave or an order processing system where customers can order items by specifying the attributes of the specific application. Web applications then can be modified by changing the values for the attributes of the meta-model. As will be explained later in this chapter the meta-model is organised into three abstract levels: shell, applications and functions. The aspects common to all Web applications such as navigation and access control are modelled at shell level. The aspects common to specific Web applications such as workflows are modelled at application level. The requirements specific to a views or user interfaces required to perform actions in an application are modelled at the function level. The meta-model presented in this thesis is implemented by extending the Component based eApplication Development/Deployment Shell (CBEADS). CBEADS is briefly described in section 1.2.3 CBEADS framework consists of engines that can instantiate the Web application from the
specifications (A. Ginige, Liang, Marmaridis, A. Ginige, & De Silva, 2007). CBEADS can support creating different types of Web applications.

The end-users should be provided with a set of meta-design tools to create or modify the Web applications. These tools should use end-user development concepts and provide easy to use interfaces which include WYSIWYG editors, drop down menus. The amount of computer domain knowledge required by the people developing the application can be minimised by developing appropriate meta-models to conceptualise the applications users want, and by providing suitable tools to support the creation of the applications as an instance of the meta-model. The activities required to specify the attribute values in the meta-model to create or modify the Web applications have different levels of complexities. The activities can be categorized into three complexity levels called routine level, logical level and programming level.

- Routine level: consists of activities where users can feed direct data from the problem domain without much manipulation.
- Logical level: consists of activities that require end-users to formulate the problem domain concepts in a given format and derive data required to populate the meta-model attributes.
- Programming level: consists of activities that require users to code the aspects.

The properties of the meta-model and the different complexity levels help end-users to participate in Web application development at different levels. For example, they can reuse a template available in the framework and concentrate on creating the content for the Web site. If they want a different look and feel, but if they do not have the required skills to
accomplish it, then the meta-model will help end-users to identify the different activities they need to seek help from professional developers. Such a socio-technical environment can eventually lead to end-users being able to develop complete applications.

This chapter presents the meta-design paradigm which can be used by end-users to develop different types of Web applications. First the meta-model is explained in detail including the meta-model properties. Then two meta-models for process centric Web applications (which cover both data intensive Web applications and Workflow intensive Web applications) and information centric Web applications are presented. Next three complexity levels of the activities required to create or modify the applications using the meta-model are discussed. After that the meta-model based meta-design paradigm is explained in detail. This is followed by a discussion on how the meta-design paradigm can address the end-user requirements presented in chapter 3 and 4.

5.2 Meta-Model

Business end-users require different Web applications for proper operation of their organisations. Therefore, the meta-model is developed to support a set of Web applications rather than just one application. SMEs require different applications to support their business processes. The shell level of the meta-model is provided to hold the different applications required by the business. In other words the shell becomes the container for the applications. In the meta-model, the common aspects required to model many Web applications are modelled at the shell level.
An application supports specific business processes within an organisation. For example, the leave processing application would support Human Resource Management and the order processing application would support the sales process. The common aspects of an application are modelled at the application level of the meta-model.

An application is viewed as a collection of use cases which supports a specific business process. For example a leave processing application can have use cases such as apply leave, approve leave, and view leave history. Use cases deliver specific functionality of the application to the users. The use case specific aspects required to implement the functionality such as interfaces are modelled at the function level of the meta-model. Three levels of the meta-model are shown in Figure 5.1.

![Three level Meta-model](image)

**Figure 5.1: Three level Meta-model**

As mentioned previously a Web based application is viewed as an instance of the meta-model. The relation between a model and its meta equivalent is similar to that between an instance an a class (Mili, Pachet, Benyahia, & Eddy, 1995). An example of an instance of the meta-model for workflow intensive Web applications is a leave processing system where employees can apply for leave. Meta-model elements are conceptual level aspects such as
user, role, form, user interface, and business object. Theoretically the application can be generated by specifying the values for the attributes of an appropriate meta-model providing a framework with relevant components which can execute the functionality of the application.

A meta-model has to cater for the different levels of capabilities of end-user developers. For example, end-user developers may not have the expert knowledge to specify the template of the Web site. However, if the end-user is provided with a template he may be able to specify the content of the site. In some cases end-users may want to develop specific templates for their Web sites. The properties of the meta-model, inheritance and overriding, which are described in next section solve this issue.

5.2.1 Meta-model properties

This section explains two properties of the meta-model called inheritance and overriding followed by a discussion on how these can solve issues with different levels of capabilities of end-users.

1. Inheritance: The models at a lower level can inherit the models at an upper level. For example an application by default inherits the presentation style defined at the shell level. Functions can use the UI definitions modelled at the application level. This property allows end users to use the existing features of an application with less effort.

2. Overriding: At a lower level the inherited properties can be overridden. For example at the application level we can override the default presentation style inherited from the shell. So each application can have its own presentation style.
These two properties provide a balance between flexibility and ease of use. For example, an end-user developer can develop a Web application with a default presentation style inherited from the shell. He/she doesn’t have to bother about the presentation styles and templates at the application level. However, if a custom look and feel is required the end-user developer can override the default presentation style provided at the shell level in the application level.

5.3 Meta-model of Process-centric Web applications

Process-centric Web applications have a common pattern “input-store-process-report information”. The users need to input the information to be stored, process information and report information in a process centric Web application. Workflow intensive Web applications and data intensive Web applications are grouped into this category. For example, in a workflow intensive Web application, one user starts the process by submitting the information. Then the information may flow from one person to another in a business process such as a leave form being forwarded to the manager for approval. Users want to report the submitted and processed information. Thus they follow the input, store, process and report information pattern. In data intensive Web applications users want to input store process and report information. For example, in a product catalogue administrator wants to store the product information, and the customer wants to process an order. Once the customer submits the order, the order data may be stored and e-mailed to the administrator. The administrator may view the order data. Therefore the meta-model of process-centric Web applications at the conceptual level consists of forms being routed based on rules, storing information entered through forms, and producing reports. Section 5.3.1 presents
the mapping of the conceptual model aspects into the meta-model. The following sections explain the meta-model of process centric Web applications in detail.

5.3.1 Mapping of aspects from conceptual model to meta-model

![Diagram of conceptual level and meta-model mapping]

**Figure 5.2: Mapping of process centric Web applications from conceptual level to the meta-model**

Figure 5.2 presents the mapping of the aspects of the conceptual level of process centric Web applications to the meta-model. Process centric Web applications such as “Leave Processing System or Purchase requisition system” which are involved with a workflow have abstract concepts such as forms, reports, procedures, users and sequence of actions. For example, as shown in Figure 5.2 a leave processing system consists of leave forms, reports, leave applicant, manager and Human Resource officer as users and the form needs to be forwarded to the relevant users after being submitted to the system. To model this at the
conceptual level, aspects such as presentation, navigation, access control, composition, workflow and data are required. These aspects are stored and handled at different levels in the meta-model. In addition to these aspects the end-users need functions to manage the applications. Application management within the framework happens at the shell level.

The meta-model for process-centric Web applications can be defined as follows.

- Shell Level: Aspects common to many process-centric Web applications such as user, navigation are modelled at this level.

- Application Level: Aspects common to a process-centric Web application such as workflow are modelled at this level.

- Function level: The function specific aspects required to implement the functionality are modelled at this level. Examples are user interface model UI and action model.

The meta-model for process centric Web applications is shown in Figure 5.3.
5.3.2 Shell level

Common functionalities required in most of the process-centric Web applications are identified by analysing many process centric Web applications at the conceptual level. These common functionalities are executed at the shell level. User model, access control model, navigation model and business object model are four models stored at this level. The shell level of the meta-model is shown in Figure 5.4. Each user has a profile. A profile can have many properties such as name, address, e-mail address, etc. A user can belong to many groups. A group can have group attributes. A group can have one or more users. Each group has functions in an application assigned to it. An application has presentation properties. Each function associates with a menu link. Business objects have many attributes and associations.
5.3.2.1 User model

There are users/participants who belong to different groups in any Web application. A group has group attributes. The user model defines the user’s profile and the groups the user is assigned to in a business Web information system. The user model is used to authenticate the users. At minimum the properties user name, password, first name, last name and e-mail are stored in the user model.

5.3.2.2 Function level access control model

Every process centric Web application has persistent functions which users want to use. Users are authorized to perform the functions based on the function level access control model. For example the “Leave Processing Application” includes a function for an employee to submit a leave application. The function level access control model is used to authorise
the authenticated users to access functions. The function level access model defines the function of the applications and the groups that can access the functions.

5.3.2.3 Navigation model

The navigation model is the mechanism that authenticated users can use to access the authorised functions. In relation to the navigation model there are two types of functions in process-centric Web applications called state independent functions and state dependent functions. In the leave processing system persistent functions like “apply for leave”, “view leave history”, which are always available to authorized users, are examples of state independent functions. These functions are independent from the state of any processes. Once a user logs in, the user will be provided with a menu to access state independent functions based on the navigation model. On the other hand, the “approve leave application” and “process leave application” are examples of state dependant functions. These functions are available to users only if a leave form is waiting for approval or processing for that particular user at that time. These functions in all applications can be accessed via a menu called “My Tasks” at one place. The navigation model defines the menu link and the navigation type for functions.

5.3.2.3.1 Common Business Object Model

Applications need to share data between them. The common Business Object model defines shared data in process centric Web applications. For example, business objects such as ‘employee’, ‘products’, which are used in many applications are kept at shell level. In other words the common data repository is managed in the shell. Business objects define attributes and relationships between objects.
5.3.3 Application level

As mentioned earlier an application consists of many functions. Therefore, an application consists of models which support many functions. The application level of the meta-model is shown in Figure 5.5. An application inherits the function level access control, common business objects and navigation models from the shell level. It consists of a workflow model, an instance level access model and application specific object models.

![Diagram](image)

Figure 5.5: Application Level of the meta-model

5.3.3.1 Workflow model

In a process-centric Web application, the business rules govern what happens next when a function is performed. In the leave application example, when an employee submits a leave form a link to access the function to approve the leave should become visible to the manager. Conditional rules can also be associated with the flow. For example, there can be a rule to say leave applications for a period of more than 3 months should be approved by the
director of the company. Once it is approved by the head of the department it should get
directed to the director instead of the human resource division. The flow is defined in the
workflow model at the application level. The workflow model defines the state, entry
condition, exit condition, transition, transition activity, exit condition. Transition is also
associated with a function and UI action.

5.3.3.2 Instance level access model

When business objects are accessed through the functions there could be rules specifying
who can access what instances of a business object. For example, if the organisation has
different departments such as sales division, production division, and accounting division
then it may be necessary to specify that the leave form from an employee in a particular
division needs to be approved by the manger of that division. The ‘project team’ that
participates in actions in functions in the workflow can be identified by applying instance
level access rules. The instance level access control model specifies the project team, the
members of the team and the functions they can access at a given time.

5.3.3.3 Application specific business objects

The application may have business objects specific to the application. Examples of
application specific business objects are reference data objects used in an application such as
leave types. That means these data may be used only within the specific application.
However, if the user wants he can store them at shell level.

5.3.4 Function level

Functions are the way of performing the actions in an application. The UI is the mechanism
used to perform the functions. UIs, business rules associated with UI elements in the input
mode and UI Action models are defined at the function level. The function level meta-model is shown in Figure 5.6.

![Diagram of the function level meta-model](image)

Figure 5.6: Function level of the meta-model

### 5.3.4.1 User interface model

The user interface model defines the interfaces used in functions of the application. The user interface model consists of UI guide, UIElementGroup, and or UIElements, and UI Actions. UI elements can be in input mode or output mode. UI elements of a form are in Input mode. UI elements of a report are in output mode. UI_Guide provide the guidelines for using a particular interface. For example, in a form interface, the guidelines can help the users to understand the purpose of the form UI. In a report interface, the guidelines can help the users to interpret the report properly. UI is associated with a business object which binds with the interface. Help tips can also be associated with the UI element in input mode. Help tips help users to fill in the values of the form UI element correctly. Sometimes, it is
necessary to logically group UIElements. For example a product order may include more than one product. The data required for each product order can consist of quantity of product and price of product. Thus, product details can be in a UIElement group of the order UI. UI Actions such as add product, amend product actions can be associated with that UIElement group. Order UI model can have UI action to process the order.

5.3.4.2 Logic model

Two types of business rules are modelled at the function level. One type is the business rules used to derive new object attribute values based on existing object attribute values. For example an organisation might give discounts based on quantity purchased; such as 5%, 10% and 15% for 10, 100 and 1000 items purchased. Thus the new object attribute total cost can be derived based on the base price, quantity purchased and applicable discount. The other category is the validation rules applied over values of the form field in a user interface. An example of a validation rule is that quantity must be a number.

5.3.4.3 UI action model

The user actions in the UI model are also modelled at function level. Examples of actions that can happen in an action model are state updates in the workflow or updates to a business object.
5.3.5 Mapping the conceptual model of process-centric Web applications to the implementation

This section explains how the specifications of the aspects in a Web application are mapped to the implementation using CBEADS framework. CBEADS framework consists of tools to support the creation of the application and engines to execute the functionality of the application as shown in Figure 5.7. Appendix A-E presents the tools and engines in detail. The specifications of the aspects for the application are stored using tools in the meta-data. This meta-data is used to create a physical perl file to be executed or to get the input for engines during the execution. The files specific to the application are stored in a separate folder.

The WMS tool, SBO tool, interface renderar and CBEADS shell consist of interfaces that support specifying the aspects of a process intensive Web application. The WMS tool provides a form based interface to specify the workflow of an application. The workflow specifications are stored in the meta-data. The SBO tool supports specifying the business objects using SBOML. The SBO tool creates the objects physically and stores the specifications in the meta-data. The interface renderar provides a form based interface to specify the interfaces. It will generate a perl file for the interface and also store the specification in the meta-data. The CBEADS shell provides form based interfaces to specify the navigation, user and access control aspects. The data are stored in the CBEADS namespace (physical database named CBEADS). This information is used to render the menu and home page. Once a user provides his/her user name and password the security component in the CBEADS shell will authenticate the user and render the home page and
menu to access the authorised functions within the applications available in the shell. The CBEADS shell uses default templates available in the Web folder to generate the pages. When a user accesses a function, then CBEADS main will execute the relevant perl file available in the specific folder for the application. Some functions that can be accessed through the menu are the start up functions of workflows. These start up functions initiate the workflows. In that case the workflow engine will execute the actions that have been triggered from the state change. Then it will pass the updates to the access data for the new state to VTMAC. VTMAC will update the activity register which will be used to manage the instance level access control. The CBEADS shell provides access to the state dependant functions through a specific page called “my task”. It will use the information provided by VTMAC to identify the state dependant functions and objects that can be accessed by the user. Next it will update the state data for the workflow instance.

Figure 5.7: Mapping the Aspects of a process centric Web application to the implementation
5.3.6 Functions to manage process-centric Web applications

The set of functions required to specify and / or modify the attributes of different aspects using CBEADS tools in process centric Web applications is given in Table 5.1. Some functions are required to manage both workflow intensive and data intensive applications while functions to manage the workflow aspect are specific for workflow intensive Web applications.

Table 5.1: Functions to manage Process Centric Applications

<table>
<thead>
<tr>
<th>Aspect</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Presentation</td>
<td>Select template and style.</td>
</tr>
<tr>
<td></td>
<td>Design the template and styles using forms and visual tools.</td>
</tr>
<tr>
<td></td>
<td>Code the styles, templates.</td>
</tr>
<tr>
<td>Data</td>
<td>Populate the values for data objects.</td>
</tr>
<tr>
<td></td>
<td>Specify the objects &amp; relationships using form or natural language.</td>
</tr>
<tr>
<td></td>
<td>Specify objects and relationships using SQL.</td>
</tr>
<tr>
<td>User (Access control &amp; personalisation)</td>
<td>Add user.</td>
</tr>
<tr>
<td></td>
<td>Update user profile.</td>
</tr>
<tr>
<td></td>
<td>Assign users to groups.</td>
</tr>
<tr>
<td></td>
<td>Assign functions to groups.</td>
</tr>
<tr>
<td></td>
<td>Populate different project teams.</td>
</tr>
<tr>
<td>Task</td>
<td>Select the operations on an object.</td>
</tr>
<tr>
<td></td>
<td>Write action display messages.</td>
</tr>
<tr>
<td></td>
<td>Specify the task using a template based form.</td>
</tr>
<tr>
<td>Workflow</td>
<td>Code the task.</td>
</tr>
<tr>
<td>----------</td>
<td>----------------</td>
</tr>
<tr>
<td></td>
<td>Modify/Create state table.</td>
</tr>
<tr>
<td></td>
<td>Create templates for the workflow actions such as e-mail.</td>
</tr>
<tr>
<td></td>
<td>Code the flow.</td>
</tr>
</tbody>
</table>

### 5.4 Meta-model for Information centric Web applications

Information-centric Web applications have the common pattern “create-manage-publish information”. In these Web applications users need to create the Web pages, manage the Web pages and publish information. Therefore the meta-model of information-centric Web applications at the conceptual level are site structure, Web pages and content. Section 5.4.1 presents the mapping of the conceptual model aspects into the meta-model followed by the details on the information centric Web applications.

### 5.4.1 Mapping of aspects from conceptual model to meta-model

Mostly SME end-users host the Web sites of their businesses with Internet Services Providers (ISPs). However, they want to update their Web sites with up-to-date information. A Web site can be abstracted to pages, content, site structure at the conceptual level. Web sites follow the pattern “create-manage-publish information”.
Figure 5.8: Mapping the aspects from Conceptual model to the meta-model

The meta-model for information-centric Web applications is shown in Figure 5.9. The hierarchical meta-model for information-centric Web applications can be defined as follows.

- Shell level: Aspects common to many information-centric Web applications such as user, and navigation which are required to manage the content of the information centric Web applications are modelled at this level.

- Application level: Aspects common to an information-centric Web application such as site structure, pages and content are modelled at this level.
5.4.2 Shell level

Common functionalities required in most information-centric Web applications are identified by analysing many information centric Web applications at the conceptual level. These common functions include the functions to manage the Web sites. The user model, access control model, navigation model and meta-Web site object model are four models stored at this level. The shell level of the meta-model is shown in Figure 5.10. User, access control and navigation models for information centric Web applications are the same as discussed in section 5.3.2. The new function is used to manage and publish Web sites. A user who is authorised to access the functions related to content management can create, manage content and publish it. The information related to the site such as name, owners user id and hosting details are stored at shell level.
Figure 5.10: Shell level in meta-model for Information centric Web applications

5.4.3 Application level

A Web site consists of many pages. Therefore, the aspects required to model a Web site are stored at the application level. The application level of the meta-model is shown in Figure 5.11. Pages inherit the presentation properties from the shell level. But, it can override at the page level. Page content and page level access control data are stored at this level.

Figure 5.11: Application level in meta-model for Information centric Web applications
5.4.4 Mapping the conceptual model of Information-centric Web applications to the implementation

This section explains how the meta-model of information-centric Web applications is mapped to the implementation using the CBEADS framework. Appendix B-E presents the tools and engines in detail. The CMS tool provides the functions create site, create page, publish tool, WYSIWYG editor, create/edit menu data, template selector and template editor to create and update an information centric Web site as shown in Figure 5.12. Create site function allows the user to create a Web site. It creates the folder structure (which contains sub folders to store images and template files) to store Web pages and associated files. It also copies default template files to be used to generate pages of the site and creates the data structure to store the conceptual model of the Web site in the database. This data structure helps to change and regenerate the Web pages. Create page function allows new Web pages to be created. It updates the page info data and creates the Web page in the folder of the Web site using the templatetoolkit (A perl library which can be used to create HTML pages) which is available through the CBEADS infrastructure. The WYSIWYG editor provides a word like interface to add/ delete/edit content of the Web pages. The create/edit menu tool allows specification of the order of the other pages to be accessed from the home page of the site. These data are stored in the database and used to generate the HTML pages. The template selector helps to change the default template of the Web site. The template and CSS editors allow editing the template and CSS file. The publish tool allows users to specify the FTP connection data and transfer the files in the folder to the ISP server.
Figure 5.12: Mapping the Aspects of an information-centric Web application to the
implementation

5.4.5 Functions to manage information-centric Websites

The set of functions required to specify and/or modify the attributes of different aspects in
information-centric Web applications using CBEADS tools is given in Table 5.2.
Table 5.2: Functions to manage Information-centric Web applications

<table>
<thead>
<tr>
<th>Aspect</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Presentation</td>
<td>Select template and style.</td>
</tr>
<tr>
<td></td>
<td>Design the template and styles using forms and visual tools.</td>
</tr>
<tr>
<td></td>
<td>Code the styles, templates.</td>
</tr>
<tr>
<td>Content</td>
<td>Add/modify content in a page.</td>
</tr>
<tr>
<td></td>
<td>Publish content.</td>
</tr>
<tr>
<td>Primary Navigation</td>
<td>Change the order in menu.</td>
</tr>
<tr>
<td></td>
<td>Specify the menu using a form.</td>
</tr>
<tr>
<td>Access Control</td>
<td>Add user.</td>
</tr>
<tr>
<td></td>
<td>Update user profile.</td>
</tr>
<tr>
<td></td>
<td>Assign users to roles.</td>
</tr>
<tr>
<td></td>
<td>Assign functions to groups.</td>
</tr>
</tbody>
</table>

5.5 Complexity Levels

The functions that support instantiation and modification of the instances in the meta-model can be grouped into three complexity levels. Some of these functions can be performed at runtime of the application while other functions can only be performed during the design time of the applications. This grouping helps end-users to identify the development activities or maintenance activities they can do based on their modelling and programming knowledge. The three complexity levels are routine level, logical level, and programming level.
The complexity levels also help end-users to employ different end-user technologies appropriately at different levels. A manual providing guidelines and programming by demonstration can be grouped into routine level activities. The logical level activities can be supported with visual tools. For example, the UI modelling tool should provide a preview of the interface being built. Visual programming is a technique used for end-user development at this level. Programming by example can be used at programming level to support coding.

The following sections explain these levels. Table 5.3 shows the categorization of the meta-model activities related to different aspects into these different complexity levels.

### 5.5.1 Routine level

Routine level activities support day to day operation of the Web information system. These can be performed during the runtime of the system. Shell level activities such as manage user model, manage navigation model and function level access control belong to the routine level. Application level activities such as populating application specific business objects and instance level access control are grouped into the routine level. Function level activities such as adding help information to the interfaces and selecting object attributes to be displayed on interfaces also are categorised as routine activities. The user can perform these activities by following an instruction manual.

### 5.5.2 Logical level

The logical level consists of activities that require end-users to analyse requirements and formulate domain knowledge in a given format. These activities are performed at the design time of the application. At design time end-users populate the meta-model to create the
application instance they want. If users want to change the application after they have started using the application they have to switch to design time and perform these changes. Some changes at this level may need to be supported by some programming level activities. However, it depends on the nature of the change done at the logical level. For example, in the workflow model if a rule called “send the e-mail to person X” is added, then a new activity “send e-mail” should be defined along with the parameters. In this case any coding is not required provided that a ‘send e-mail’ function exists. However, if a send e-mail function is not available then it needs to be coded. Creating business objects and creating workflow activities at application level are logical level activities. Also creating user interface definition at function level is a logical level activity.

5.5.3 Programming level

The programming level consists of activities that require users to model and code to support the requirements model at the logical level. These activities are done at the design time. Coding the user interface actions, logic for validation and object value inference rules belong to the programming level.
Table 5.3: Meta-model activities at different complexity levels

<table>
<thead>
<tr>
<th>Complexity Aspects</th>
<th>Routine</th>
<th>Logical</th>
<th>Programming</th>
</tr>
</thead>
<tbody>
<tr>
<td>Presentation</td>
<td>Select template and style $[I,P]$</td>
<td>Design the template and styles using forms and visual tools. $[I,P]$</td>
<td>Code the styles, templates. $[I,P]$</td>
</tr>
<tr>
<td>Composition</td>
<td>Add/modify content in a page. $[I]$ Add help, UI guide and help tips. $[P]$</td>
<td>Change the label Specify forms and tables using a form. $[P]$</td>
<td>Code the forms and tables using HTML $[P]$</td>
</tr>
<tr>
<td>Primary navigation</td>
<td>Change the order in menu. $[I]$</td>
<td>Specify the menu using a form. $[I]$</td>
<td></td>
</tr>
<tr>
<td>Data</td>
<td>Populate the values for data objects. $[P]$</td>
<td>Specify the objects &amp; relationships using form or text. $[P]$</td>
<td>Specify objects and relationships using SQL $[P]$</td>
</tr>
<tr>
<td>User (Access control and)</td>
<td>Add user, update user profile, Assign users to roles, Assign</td>
<td>Define project team templates using a form. $[P]$</td>
<td></td>
</tr>
<tr>
<td>personalisation</td>
<td>functions to groups. ([I,P])</td>
<td>Define the user attributes using a form. ([P])</td>
<td></td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------</td>
<td>--------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Populate different project teams. ([P])</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Task</th>
<th>Select the operations on an object. ([P])</th>
<th>Specify the task using a template based form. ([P])</th>
<th>Code the task. ([P])</th>
</tr>
</thead>
<tbody>
<tr>
<td>Write action display messages.([P])</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Workflow</th>
<th>Create/ modify state table. ([P])</th>
<th>Create templates for the workflow actions such as e-mail. ([P])</th>
<th>Code the flow([P])</th>
</tr>
</thead>
</table>

I-Meta model for Information Centric Web Applications

P – Meta model for Process intensive Web applications
5.6 Meta-model based meta-design paradigm

The meta-design paradigm is focused on providing a technical infrastructure to develop open ended systems that allow end-user developers to evolve the systems, providing a learning environment which supports end-users to learn and be involved in activities required to create / maintain Web applications, and providing a socio-technical environment that allows end-users and professional developers to collaborate in development and maintenance of the systems. This section explains how the meta-model based meta-design paradigm provides infrastructure, a learning environment and a socio-technical environment to support end-user development of Web applications.

5.6.1 Infrastructure

The design of the infrastructure, in a meta-model based meta-design paradigm should support end-user development activities such as customisation, end-user modifications, and end-user programming which allow end-user developers to create/ modify the systems. Granular, malleable, and modifiable computer systems are the prerequisites that enable the system and users to evolve together (Giaccardi, 2003).

A meta-model based infrastructure provides an open system which allows end-user developers to design and modify the systems to support evolving requirements. End-users can instantiate the required Web applications in the meta-model by specifying the attributes of the aspects at the conceptual level. For example, the meta-model of workflow intensive Web applications can be instantiated to create a leave application system where the employees can apply for leave or an order processing system where customers can order items by specifying the attributes of the specific application. Web applications then also can
be modified by changing the values for the attributes of the meta-model. Form based interfaces are used for the functions to manage the meta-model.

CBEADS, the infrastructure that supports the meta-model based meta-design paradigm also consists of a function called “create function” which allows the systems to add more functionality to an application (A. Ginige, 2002). The user management application provides form based interfaces which can be used to change the attributes of the aspects such as user, group, primary navigation, and access control. As explained in Appendix A the runtime programming environment provided with CBEADS allows users to switch between the development and runtime environment in a mouse click. In summary, the meta-model is implemented using an evolvable Web infrastructure which supports end-user development concepts and activities which were explained in chapter 2.

5.6.2 Learning environment

In a meta-design paradigm a learning environment should be provided to allow gradual transition of end-users from passive consumers to users, and power users (Fischer, Ye, Sutcliffe, & Mehandijev, 2004). In other words, the end-user development environment should be a gentle slope system that provides a gradual learning environment for end-users.

The different complexity levels of the meta-model activities help end-users to participate in Web application development at different levels based on their knowledge and skills. For example, they can reuse a template available in the framework and concentrate on creating the content for the Web site. Then if the end-user developers want to perform advanced development activities, they can learn and customise the templates of the Web site using a form based interface. If they want to create their own template they can write the template
files, CSS style sheets using an editor. The complexity of these three activities ranges from routine level to programming level complexities. However, even if end-users can not advance to activities with the complexity of the programming level they have the opportunity to achieve the desired appearance of the systems as explained in the next section.

5.6.3 Socio-technical environment

A meta-design paradigm should create an environment to support users to collaborate and participate in development of the Web applications. As discussed in previous sections the infrastructure and learning environment created with the meta-model based meta-design paradigm provide end-user developers with an opportunity to actively engage in development of Web applications. However, they are not limited by their capabilities or skills or other factors from using the meta-design paradigm to get the application developed according to their requirements. This is an important requirement identified in the survey with SMEs as explained in section 3.3.

The properties of the meta-model support users to get an application developed according to their requirements without constraints resulting from their level of knowledge or skills. For example, end-user developers may not have the expert knowledge to specify the template of the Web site. However, if the end-user is provided with a template, he may be able to specify the content of the site. In this situation they can use the templates provided at the shell level. The whole application inherits the presentation properties specified at the shell level. In some cases users may want to develop specific templates for their Web sites. In that case they can override the default template and use the presentation styles provided at the shell level. If the end-users can not perform this activity themselves they can get help from professional developers. The properties of the meta-model together with the complexity levels of activities support collaborative development according to the requirements of the
end-users. For example, they can reuse a template available in the framework and concentrate on creating the content for the Web site. If they want a different look and feel, but do not have the required skills to accomplish it, then the meta-model will help end-users to identify the different activities where they can get the help of professional developers. Such a socio-technical environment can eventually lead to end-users being able to develop complete applications.

5.7 Supporting the requirements for end-user development

This section discusses how the requirements of business users, particularly the SMEs, are met in the meta-model based meta-design paradigm.

5.7.1.1 Support for different types of Web applications required by SMEs

Meta-models for different types of Web applications are created based on the common patterns appeared in each type of Web applications. Section 5.3 and 5.4 presented the meta-model for the main types of Web applications: information centric and process centric Web applications. Then a set of functions required to manage the aspects in each type of Web application was identified. The instantiation of these two types of Web applications are supported by engines in CBEADS framework. Engines in CBEADS framework are the components that support the runtime operation of the application (J. A. Ginige, De Silva, & A. Ginige, 2005). For example the Workflow Engine (WFE) (Liang, Marmaridis, & A. Ginige, 2007) and Virtual Team based Access Control (VTMAC) (Marmaridis & A. Ginige, 2007) are two engines that support the process intensive Web applications. The applications
are specified using tools in CBEADS. UI Renderar, Workflow Management System (WMS), Content Management System (CMS) and User Management Systems have tools that support specifying the Web applications. Appendix A summarises the tools and engines within the CBEADS framework.

5.7.1.2 Need to support the specification of Web applications at the conceptual level

A meta-model uses the aspects of the Web applications at the conceptual level to create or modify different types of Web applications. The conceptual level is the solution to the problem that Web application attempts to solve at the application domain (Refer to section 1.2.2). Rode’s study on the mental model of end-user developers also shows that end-users use the concepts of the application domain (Rode, 2005). The study of end-user specifications in chapter 4 shows that end-users who use the aspects at the conceptual level are closer to the end-user mental model of Web applications rather than those who use the implementation domain aspects.

5.7.1.3 Common data repository

This requirement is supported by creating a common data repository which can be accessed by the applications within the shell. Business object definitions and relationships within the application will be created/modified during the development time. But these business objects are stored at the shell level. Therefore all the applications with in a shell can use the same business object thus avoiding any duplication. Shell/application becomes the name space for the business objects. The common data repository also helps to manage the data easily. Therefore data management operations such as back up, recovery are available at the shell level.
5.7.1.4 Common login to all applications within the organisation

Having all the applications within a shell facilitates maintaining a common login. The basic user object has user name, password name and e-mail address at this level. Name and e-mail address are used by the shell to communicate with the user. For example, if a user requests password retrieval, it needs to happen at the shell level and the shell uses the e-mail address to communicate with the user. A user object is a special type of object with predefined template consisting of basic user attributes such as user name, password, first name, last name and e-mail.

5.7.1.5 Balance between DIY and professional developer

The activities required to create and / or modify applications using CBEADS are grouped into complexity levels. The complexity levels help end-user developers to identify the group of activities in which they can participate. The meta-model provides default models of these aspects to assist end-user developers at shell level. For example, if an end-user wants, he/she can develop a Web application with the default presentation style provided in the meta-model. That means application can inherit the attributes defined at the shell level in the hierarchical meta-model. On the other hand shell level attributes can be overridden at the application level. For example, if an end-user wants a custom look and feel he may get a professional developer to do that for him. Inheritance and overriding at different levels help end-users to trade off between DIY and the use of a professional developer.

5.7.1.6 Support to create Web applications with specifications of minimum aspects

The meta-model supports a minimalism approach that allows end-users to create Web applications using the minimum aspects. Inheritance property of the meta-model helps to create the Web applications using the minimum aspects. This property of the meta-model
supports a minimalism approach that allows end-users to create Web applications using the minimum aspects. For example, end-user developers can specify the tasks and workflow to get the desired functionality without worrying about the look and feel. They can use the default presentation styles provided by the shell for their applications.

## 5.8 Conclusion

This chapter presented the meta-model based meta-design paradigm that can enable business end-users to develop Web applications. Most importantly the design of the meta-model has addressed the requirements of an end-user development approach suitable for business users. Creating the meta-model to support different types of Web applications will help end-users to develop different types of Web applications within a familiar environment rather than having to learn different environments to develop different applications. This will help end-users to develop their Web applications in an efficient and effective way.
6 Evaluation of Meta-model based Meta-design Paradigm

6.1 Introduction

Chapter 5 presents the meta-design paradigm based on the meta-model of Web applications. Chapter 5 also discusses theoretically how the meta-design paradigm can support business end-users to develop and / or maintain Web applications. The meta-model instantiates different types of Web applications based on the values provided for the attributes of aspects at the conceptual level. A CBEADS framework (A. Ginige, Liang, Marmaridis, A. Ginige, & De Silva, 2007) was extended to capture the attributes of the aspects required to specify information centric Web applications and process intensive Web applications. The objective of the extension to CBEADS was to facilitate a meta-design paradigm.

A series of qualitative studies was conducted to practically evaluate the meta-design paradigm. The first study was carried out with four SME tool makers who wanted to have a Web presence for their companies. Initially, the Web sites were developed for the tool makers by the students as part of a project subject. A Content Management System (CMS) tool built on CBEADS was used in these four projects. Then the tool makers were provided with the Web sites and the CMS tool. They were provided with a user-training on the CMS tool. Tool makers were involved in activities with routine level complexity such as maintaining the content in their sites. For more complex activities
such as changes to the templates, they needed help from the research group. The study evaluated the success of the meta-design paradigm based on the feedback survey. The findings of this study showed that the meta-model based meta-design paradigm can be used to manage the evolution of Web applications.

A second study was conducted with a group of Business degree students enrolled in a Business Information Systems Project B subject in University of Western Sydney to demonstrate how the meta-design paradigm can be used in developing data-intensive Web applications. The background study of the students revealed that they had no or little knowledge in Web application development. This implies that there’s no significant difference between the group of students and the business users. The study was planned to see whether students can create Web applications by providing the minimum aspects at the conceptual level. The success of end-user development is measured based on completion of the given assignments to the students. The study revealed that end-users can create Web applications using the meta-model by providing essential aspects at the conceptual level.

The third study was organised as an action research cycle carried out with a group of Masters Degree students enrolled in a Web Engineering subject. The students had formal education in IT. However, the background study revealed that they are novices in Web application development. The meta-design paradigm was used to facilitate the students to learn conceptual model aspects and develop the Web applications. This study reveals that the meta-design paradigm can also help novice developers to model and develop Web applications.
This chapter presents the three studies demonstrating the meta-design paradigm. First, the case study on tool makers is presented. Then the case study on Business Degree students involved in the project subject is described. Next the action research study conducted with the students involved in a Web Engineering subject is described. After that the strategies derived, based on the studies, that could be used with the meta-design paradigm are presented. Finally the chapter concludes with the lessons learned from these studies.

6.2 Study 1: Tool makers project

This study was a part of a bigger eCollaboration project which was organised by Austool Ltd at Ingleburn (Arunatileka, A. Ginige, Hol, & Lawson, 2005). A group of four SMEs in the toolmaking industry, with limited ICT experience and no Web sites participated in this study. The overall aim of the research project was to conduct a pilot study to investigate how the toolmaking industry in Australia can benefit from advances in information and communication technology to become globally competitive. The specific aim of this research study was to provide SMEs with a basic Web site and a tool to maintain the Web applications. This was a required step in the first phase of the eCollaboration project according to the eTransformation road map (Arunatileka, A. Ginige, Hol, & Lawson, 2005) (eTransformation roadmap was discussed in section 2.5.1). Figure 6.1 also highlights the interactions possible at the business-to-customer level through the Web site (Arunatileka, A. Ginige, Hol, & Lawson, 2005).
6.2.1 Method

First a SWOT analysis (Analysis of Strength, Weakness, Opportunity and Threat) of the SMEs was conducted. This helped to identify the factors that can affect the project and to establish the position of SMEs in the eTransformation roadmap. These factors include IT infrastructure, existing IT support, time, affordability, and capabilities of SMEs related to software applications. Some of these factors were revealed in the survey with SMEs on end-user development that was presented in chapter 3. Table 6.1 describes the status of these SMEs with respect to these factors.
Table 6.1: Factors present with the tool makers

<table>
<thead>
<tr>
<th>Factor</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICT infrastructure</td>
<td>ICT was used only to support manual tasks and operations.</td>
</tr>
<tr>
<td>Existing IT support</td>
<td>None of the toolmakers has inhouse IT support staff. Two had help from their children.</td>
</tr>
<tr>
<td>Time</td>
<td>Toolmakers were working in their business while doing the office administration, marketing. Sometimes they get help from family.</td>
</tr>
<tr>
<td>Affordability</td>
<td>Toolmakers were struggling in their businesses at the time so they wanted to find a solution through collaboration.</td>
</tr>
<tr>
<td>Capabilities of SMEs</td>
<td>All four tool makers did not have formal education in IT. However, they were familiar with e-mail, word, excel, CAD CAM. None has developed Web sites.</td>
</tr>
</tbody>
</table>

The impact of these factors on the project was analysed during the feasibility study. This analysis helps to specify the guidelines to the meta-design approach as given in Table 6.2. Since the tool makers were busy with day to day business and they had no experience in Web application development, it was recommended that design and development of the Web applications be done by the AeIMS research group. The initial Web site is equivalent to the ‘seed’ in the meta-design process model explained by Fisher, G. (2005) (Meta-design process model is explained in detail in section 5.6.) Then the SMEs were provided with the CMS tool within a CBEADS framework to manage the routine level and logical level activities related to the maintenance of their Web site (The different activities that relate to information centric Web applications at routine
and logical level complexities are given in section 5.5). In other words the users of the system were provided with a tool to evolve the ‘seed’. The CMS tool is presented in appendix C.

**Table 6.2: The guidelines to meta-design approach**

<table>
<thead>
<tr>
<th>Factor</th>
<th>Recommendations to suit the impact</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICT infrastructure</td>
<td>Need to get a computer which deployed the CBEADS framework and tools.</td>
</tr>
<tr>
<td>IT support</td>
<td>Need IT support for some time until SMEs are competent in managing their applications.</td>
</tr>
<tr>
<td>Time</td>
<td>Development phase: Need to manage meetings uninterrupted during the design phase.</td>
</tr>
<tr>
<td></td>
<td>Maintenance Phase: Need to make the user interfaces of the tool closer to the user interfaces that SMEs are familiar with.</td>
</tr>
<tr>
<td>Affordability</td>
<td>Get the student groups to develop the applications so it provides a win-win situation for both toolmakers and students.</td>
</tr>
<tr>
<td>Capabilities of SMEs</td>
<td>Need to provide tools to similar to the tools that they use now.</td>
</tr>
</tbody>
</table>

Four student groups enrolled in the eBusiness course undertook the development of four Web sites. It was planned to gather the requirements for the Websites from SMEs. Meetings were scheduled early morning or at toolmaker’s convenience so their business activities were uninterrupted. The meetings between the project team and the SMEs were organised by researchers in the AeIMS research group. Members of the research
team made presentations in the meeting. This guaranteed that always the required data and design decisions were made for the betterment of the toolmakers. For example, they facilitated the meeting by showing previous Web sites developed for SMEs by the group. The researchers had to organise several design meetings to create and finalise the look and feel and structure of the Web sites. However, there were delays in specifying the requirements from SMEs. Students completed the Web sites, by the deadlines with the decided structure, look and feel and the made up content. It took longer than expected to complete the Web sites. Researchers had to help SMEs with the content and images for the Web site. After they got the initial Website developed, SMEs took the initiative to evolve the seed Website by coming up with modifications to the structure, look and feel and content. Finally the Web sites were hosted and the CBEADS framework with the CMS tool was deployed in four computers to SMEs. It was planned to integrate the eCollaboration tools into the same CBEADS framework.

Examination of the impacts and experiences with SMEs during the development of the Web sites indicates that the meta-design approach could be used during the maintenance phase which could be beneficial to the SMEs. The toolmakers were given the user training. A manual was provided. The CMS tool consists of a WYSISWYG editor to maintain the content of the site and forms to create new pages, edit menu, publish the content. The user interfaces in the tool provide the required guidelines to specify/modify the aspects.

One SME dropped out from the study due to changes to his business (relocations and change of name and structure). This shows a need for a development approach such as a meta-design approach that can support co-evolution. Even though the research group helped this SME to host his Web site, it took more time than planned so the
maintenance phase could not be monitored. The AeIMS research group helped and monitored the maintenance activities of the other three Web sites.

6.2.2 Results

The maintenance activities (i.e. the evolution of the seed Web sites) carried out during the first six months are listed in Table 6.3. It shows that SMEs have been involved in activities at the routine level. When the required activity is complex the end-users required support from the developers.

Table 6.3: Maintenance Activities

<table>
<thead>
<tr>
<th>Activity</th>
<th>Complexity of task</th>
<th>Who is responsible</th>
</tr>
</thead>
<tbody>
<tr>
<td>Updating the employment</td>
<td>Routine level</td>
<td>SME</td>
</tr>
<tr>
<td>opportunities</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Updating the product data</td>
<td>Routine level</td>
<td>SME with the help from</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Researcher</td>
</tr>
<tr>
<td>Change of banner</td>
<td>Programming level</td>
<td>Researcher</td>
</tr>
</tbody>
</table>

SMEs feedback on the easiness of the maintenance tasks is given in Table 6.4. All these tasks introduced to them are at the routine level of complexity. The feedback rating is explained below:

1: I can’t do it, it’s too difficult;
2: I needed some assistance at first, but can do it on my own after;
3: I can do it after explanation and referring to the manual;
4: I can do it myself by referring to the manual.
SMEs were confident that they can continue doing all the tasks by following the manual or with some assistance for the first time.

Table 6.4: SMEs feedback on routine level development activities relate to aspects

<table>
<thead>
<tr>
<th>Aspect</th>
<th>Function</th>
<th>Tool maker 1</th>
<th>Tool maker 2</th>
<th>Tool maker 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Presentation</td>
<td>Select template and style</td>
<td>4</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Content</td>
<td>Add/delete page; Add/modify content in a page; Publish page;</td>
<td>4</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Primary Navigation</td>
<td>Change the order in menu; Specify the menu using a form;</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Access Control</td>
<td>Add user; Update user Profile; Assign users to roles; Assign functions to</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

6.2.3 Discussion

The infrastructure for this project was implemented in CBEADS based on the meta-model of information centric Web applications. The meta-model of information centric Web applications helped the users to change the attributes of the aspects at the
conceptual level to evolve the applications according to changing requirements. The experience with toolmakers during the requirement gathering and design phase shows that it would be difficult to get the application developed without significant help from the researchers. This was mostly due to the time limitations and enormity of the task. Then the observations during the maintenance and the feedback from the toolmakers show that SMEs could evolve their Web site with initial support from the research group. This verifies that the meta-model based development approach had provided a suitable learning environment for the SMEs. Establishment of the complexity levels of the activities identifies the activities in which end-users can participate. Therefore, the meta-design paradigm can help to achieve a balance between the DIY professional developers. This helped to create a socio-technical environment of collaboration between users and developers. Reflection on the experience of the project leads to identification of the factors that affect the implementation of the meta-design paradigm. These factors include characteristics of individuals, support, and infrastructure. These factors are described in section 6.5. The next study was planned to explore the meta-design paradigm during the development phase.

The limited time available for the study made it difficult for the researcher to plan it with a group of business users for two reasons: 1) It was clear that developers have to provide a significant level of support during the development; 2) Based on the priority of the tasks for the business users, it may be difficult to complete the study in the allocated time. Therefore, the study was planned to be conducted with a group of students with a similar background to the business users.
6.3 Study 2: Student projects

The study described in this section demonstrates the meta-design paradigm used to develop information centric Web applications and data-intensive Web applications. This study was conducted with a group of Business Degree students who enrolled in a Project B subject. The group of students voluntarily participated in the study.

6.3.1 Background study

The background of the students was studied using a survey. The students’ knowledge, capability and experience in Web application development technologies were collected. The questions were designed to identify the background knowledge and skills in IT, particularly in Web application development. They were also asked to briefly explain any Web projects they had worked on before to identify how familiar they are with the different implementation domain concepts given in Table 6.5. The set of concepts was built on the set of activities Rode et al. (2004) identified as concerns of end-user developers. First these concerns were checked against the different types of Web applications to complete the full list of concepts needed in developing these applications. Then six experienced developers went through the list to identify the importance of each task related to the development of different types of Web applications.
Table 6.5: Categorisation of different Implementation Domain Concepts

<table>
<thead>
<tr>
<th>Concept Group</th>
<th>Concepts</th>
</tr>
</thead>
</table>
| 1 Infrastructure | Server configuration  
Development environment installation |
| 2 Static Web site | Layouts  
Graphics  
HTTP  
Debugging  
Usability |
| 3 Dynamic pages | Data (SQL queries)  
Reading data from a database table and display on a Web table  
Debugging  
Usability  
Concepts required for static-pages |
| 4 Forms | Data (Object modelling, relational database design, create databases)  
User input validations  
Client side scripting  
Storing form data in database  
Concepts required for static-pages |
| 5 Processes | User authentication and authorisation  
Business rules  
Concepts required for dynamic pages. |

6.3.1.1 Initial survey

Figure 6.2 summarises the students’ experience in Web application development technologies such as HTML, CSS, client side scripting, server side scripting and SQL. It
shows the percentage of students that had little knowledge, some knowledge, adequate knowledge and good knowledge in each technology. According to the chart, the majority of students (48%) had adequate knowledge in HTML, but little knowledge in server side scripting and some knowledge in CSS and client side scripting.

![Bar chart showing percentage of students' knowledge in different web development technologies]

**Figure 6.2: knowledge in different technologies used in Web application development**

The second part of the questionnaire attempted to identify the students' level of skills in different implementation domain concepts given in Table 6.5. Students were asked to rate their capability from 0-5 for the different sub-concepts which form the main implementation domain concepts given in Table 6.5. Students were also asked to describe the technologies or the development environments that they had used in dealing with these concepts. The rating was interpreted as follows: 0-don’t know; 1-very difficult; 2-difficult; 3-neither difficult nor easy; 4-easy; 5-very easy.
The mean value of the cumulative rating of difficulty for implementation domain concepts is given in Figure 6.3. The rating for the questionnaire was inversed to calculate the difficulty.

![Graph showing mean hardness for different implementation domain concepts](image)

**Figure 6.3: Mean values for Capability of using implementation domain concepts**

According to the graph, it would be very difficult for students to manage the infrastructure. However, they had little difficulty in developing static sites and dynamic Web pages. It seems they can more easily develop Web forms and dynamic pages than Web sites with graphic and styles. The reason for this was their existing knowledge of concepts associated with databases. They were familiar with the basic database concepts such as creating database and SQL queries. However, it was difficult for them to develop processes intensive Web applications. This can also be related to the limited knowledge in server side scripting in the knowledge graph (Refer to Figure 6.2).

According to the responses to the questionnaire all students had completed other subjects related to Information Technology such as Database Principles, Principles of Information Systems and Business Application Development 1 and 2 (Java).
6.3.1.2 Challenges in using CBEADS for the projects

The students had gathered the requirements and come up with the conceptual model of the applications in semester 1. The four design reports they produced in semester 1 suggest that two of the projects must be implemented as Web applications while the other two project groups can provide more benefits to their clients if implemented as Web applications.

The students were introduced to the CBEADS framework. But the students were given the opportunity to select the technology to implement the projects. The possibility of them using some other technology rather than CBEADS was acknowledged at the beginning of the study. Rosson identified that end-users are not willing to switch to other tools due to lack of expertise, time and money (Rosson, Ballin, & Heather, 2004). The first two reasons are valid in this case even though CBEADS was available freely to the students. A version of CBEADS was distributed through platform Web, where the subject materials are normally distributed in the University. However, none of the project group had installed CBEADS by the second week. Students first reported difficulty in either downloading or installation of the software. Then they were provided with the software and the installation manuals in their memory sticks assuming that the problem may be the slow network connections at their homes. Further, a CBEADS installation session was held on 5th Week for the students who were willing to use CBEADS since the infrastructure set up had been identified as a hard task according to Figure 6.3. The students didn’t take up this opportunity. They didn’t bring the laptops as promised to the class. Only one group used CBEADS for the development of the application. However, even that group couldn’t complete the project because they put little emphasis on the implementation and didn’t follow the given instructions. For example, students didn’t get CBEADS installed until 9th week of the semester. This
experience indicates that managing the infrastructure should be treated as an activity equivalent to programming level complexity.

6.3.2 Lab experiment

The analysis of the capabilities of the students with related to the development of Web applications and the challenges using CBEADS for the project indicate that management of infrastructure is difficult for the students. Therefore the following hypothesis is established: “If the students are given the infrastructure, they may be able to develop data intensive Web applications and information centric Web applications by specifying the minimum aspects at the conceptual level”. Data intensive and information centric applications were selected since those are the types of applications students had for the projects. To test this hypothesis, a server with a CBEADS environment was used. Students had only to login to the server to access the CBEADS framework. This minimised the challenge in managing the infrastructure by bringing the complexity of it to the routine level.

6.3.2.1 Participants

Seven students from the Project B class volunteered to do the lab assignment. Students were invited to the session at the last formal lecture during the semester. The session was organised during the semester break.

6.3.2.2 Method

The selected target applications were data intensive Web applications and information centric Web applications. Students were provided with a manual of application development within CBEADS. Students were given two questions to work on, one on information centric Web application and the other on data intensive Web applications. Two hours were allocated for the assignment. The assignment is given in Appendix K.
If the students asked for any help they were referred to the manual. The completion rate of the tasks was checked using the server logs. Feedback on the easiness of the tasks was collected at the end of the session. Students had to rate all the tasks they had to complete on a scale from 0-7. The rating was interpreted as 0-don’t know, 1-extremely difficult, 7-very easy.

6.3.2.3 Results

The server logs show that the students completed all the tasks. The mean value for the easiness of creating the Web applications using the meta-design paradigm versus the mean value for the easiness of creating the Web applications that the students reported at the first questionnaire was compared. The comparison is shown in Figure 6.4. Both questionnaires got the students to rate the activities required to develop the Web applications. For the comparison, only the ratings for the tasks related to developing information-centric and data-intensive Web applications given by the group of students who participated in the lab session in the first questionnaire were used. Table 6.5 shows the different implementation domain concepts related to the development of information-centric and data-intensive Web applications using CBEADS and the meta-model. Table 6.6 shows the different activities students had used in the assignment to develop the Web applications using the meta-model.

Table 6.6: Activities required in developing Web applications using meta-model

<table>
<thead>
<tr>
<th>Concept Group</th>
<th>Concepts</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Infrastructure</td>
<td>Login to CBEADS</td>
</tr>
<tr>
<td></td>
<td>Access tools for developer</td>
</tr>
<tr>
<td>2 Static Web site</td>
<td>Create new Website</td>
</tr>
<tr>
<td></td>
<td>Create pages</td>
</tr>
<tr>
<td></td>
<td>Upload images</td>
</tr>
<tr>
<td>3</td>
<td>Dynamic pages</td>
</tr>
<tr>
<td>---</td>
<td>---------------</td>
</tr>
<tr>
<td></td>
<td>Create report UI</td>
</tr>
<tr>
<td></td>
<td>Create new application</td>
</tr>
<tr>
<td></td>
<td>Add new function</td>
</tr>
<tr>
<td></td>
<td>Assign function to group</td>
</tr>
<tr>
<td>4</td>
<td>Forms</td>
</tr>
<tr>
<td></td>
<td>Creating object definitions and relations</td>
</tr>
<tr>
<td></td>
<td>Create form UI</td>
</tr>
<tr>
<td></td>
<td>Display form data</td>
</tr>
<tr>
<td></td>
<td>Create new application</td>
</tr>
<tr>
<td></td>
<td>Add new function</td>
</tr>
<tr>
<td></td>
<td>Assign function to group</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Easiness</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>100.00%</td>
<td>80.00%</td>
</tr>
</tbody>
</table>

**Figure 6.4: Easiness of creating with and with out meta-model**

### 6.3.3 Discussion

According to Figure 6.4, it is clear that the meta-model and CBEADS significantly improve the student's ability to create Web applications. Easiness in managing the infrastructure has increased from 12.85% to 97.12%. Therefore, the entry barrier to development of applications has reduced. It shows that they were able to create the information centric Web applications without much difficulty. The easiness of creating information centric Web applications has increased from 32% to 87%. However, still
they face some difficulty in creating the data intensive Web applications due to the increased level of complexity of the activities. Even in that case, reduced complexity in CBEADS and the SBO tools doubles the easiness. The results of this study show that the meta-model based development approach provides a learning environment with reduced complexity. Also it was clear that the developers should collaborate with end-user developers to ease the barriers to end-user development. However, study 1 and 2 mainly focused on evaluating two aspects, learning environment and socio-technical environment, in the development of Web applications. The next study was planned to evaluate how the meta-model based infrastructure can be used to learn modelling Web applications at the conceptual level.

### 6.4 Study 3: Action Research Cycle

The action research cycle was conducted to investigate how the meta-model can help to model Web applications at the conceptual level. The study was conducted with a group of students due to the limitated time and resources. This study also demonstrates how the meta-design paradigm can be used in the class room to help students to learn modelling Web applications at the conceptual level. 15 students enrolled in a Web Engineering subject participated in this study. An action research cycle was formulated as a practical inquiry for the study, since the study focused on interaction with participants to change the existing approach and understanding of Web application development. The other participants in the project were the lecturer-in-charge of the Web Engineering subject and the author. This study was conducted based on the canonical action research methodology (Baskerville, 1999; Bednarz, 2002; Kock, McQueen, & Scott, 1997) which includes five steps: 1) problem diagnosis, 2) action planning, 3) intervention, 4) evaluation and 5) specific learning outcomes.
6.4.1 Problem diagnosing: first step of the AR cycle

The problem to be solved in this study was “Can a meta-model based infrastructure help students to learn modelling Web applications at the conceptual level?”. The objective of the problem diagnosis was to examine the student's knowledge on the conceptual model of Web applications. The student's knowledge, capability and experience in Web application development technologies and background knowledge was collected using a questionnaire. The questions were designed to identify the background knowledge and skills in Web application development and modelling. The questionnaire is given in Appendix M. Students answered the questionnaire before any formal introduction to the conceptual modelling of Web applications. Section 6.4.1.1 presents the findings on students’ capabilities in conceptual modelling.

Most of the students had some experience in Web application development, where they had developed a Web site or data intensive Web application as an assignment during undergraduate studies. 47% had experience in Web application development for less
than 1 year (most of them say around 3 months). 33% had developed Web application for 1 to 2 years. Only 20% had experience over more than 2 years. Students had used Web application development technologies such as HTML, Java, PHP, flash, java scripts, and Microsoft access databases in their projects.

6.4.1.1 The initial findings

The answers to the questionnaires are analysed using coding techniques given in grounded theory (Grounded theory is explained in section 3.2). First the key concepts in student’s responses were identified and labelled. Then the identified key concepts were compared with the key concepts related to the aspects of the conceptual model given in chapter 4. If an answer consisted of all the key words then it was treated as a correct/matching answer. If an answer consisted of 50% key words it was treated as a close answer. If an answer consisted of less than 50% key words it was treated as a conflicting answer. The responses to the question on a definition for a conceptual model reveals that 60% of the students do not possess a clear understanding of the conceptual model even when they have some experience in Web application development. The answers to the case study on data intensive Web application reveal that about 27% of the students have a correct understanding of the presentation, access control, and tasks. Nearly 47% students had given a close answer to the data aspect while only 20% were able to provide the correct answer. 40% had some understanding on the composition model of the Web applications. It was also clear that navigation was not clearly understood compared to the other aspects. Only 7% correctly identified the navigation aspect while only 20% had a close answer for that. Most of the students did not identify navigation and presentation. This was similar to the findings with a group of administrative staff on their specifications to the Web applications.
Table 6.7: Background Study-Students responses to the conceptual model and aspects

<table>
<thead>
<tr>
<th>Aspect</th>
<th>Response</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No</td>
<td>Conflicting</td>
</tr>
<tr>
<td>Conceptual Model</td>
<td>20%</td>
<td>40%</td>
</tr>
<tr>
<td>Presentation</td>
<td>40.00%</td>
<td>26.67%</td>
</tr>
<tr>
<td>Composition</td>
<td>26.67%</td>
<td>33.33%</td>
</tr>
<tr>
<td>Data</td>
<td>6.67%</td>
<td>26.67%</td>
</tr>
<tr>
<td>Navigation</td>
<td>46.67%</td>
<td>26.67%</td>
</tr>
<tr>
<td>Access Control</td>
<td>33.33%</td>
<td>26.67%</td>
</tr>
<tr>
<td>Task</td>
<td>33.33%</td>
<td>26.67%</td>
</tr>
</tbody>
</table>

6.4.2 Action planning: second step of the AR cycle

The week by week tutorial and lecture schedule for the Web engineering course during the first 9 weeks is given in Table 6.8. The aspects required to model different types of Web applications are introduced during the lectures. Students are given some opportunity to practise modelling Web applications during the tutorials. The activities used in the evaluations of the action research are marked with *.

Table 6.8: Week by week unit schedule

<table>
<thead>
<tr>
<th>Week</th>
<th>Lecture Topic</th>
<th>Tutorial Topic/ Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Overview of Web Engineering</td>
<td>Questionnaire 1*-Background Study on student’s experience and capabilities related to Web</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>2</td>
<td>Web System Development Process</td>
<td>Tutorial 1: Web Application development environment</td>
</tr>
</tbody>
</table>
| 3 | Design Methodologies and modelling techniques | Week 3 Tutorial: Dynamic Web Application development  
**Questionnaire 2** - Modelling exercise on Web applications at Conceptual level |
| 5 | Review of Web Technologies - UI design, | Discussion on Assignment related issues |
| 6 | Web System Architectures | Week 6 Tutorial: Web Application Modelling |
| 7 | Design of Data intensive Web Systems | Presentation of Assignment 1*  
**Questionnaire 3** - choice of technology for the assignment. |
| 8 | Design of Web based Workflows | Week 8 Tutorial: Different approaches to model Web based workflow. |
| 9 | Broader Challenges in Web System Design - OCAS Case study | **Week 9 Tutorial*** - Modelling Workflow intensive Web applications at the conceptual level |
6.4.3 Action taking: third Step of the AR cycle

This section presents the application of the meta-design paradigm in the activities involved in the action research cycle in order to help students to learn modelling of different types of Web applications at the conceptual level.

6.4.3.1 Questionnaire 2

Students were introduced to the conceptual model of Web applications in Week 3. The aspects of different types of Web applications: information centric Web application and data intensive Web application, at the conceptual level were also briefly introduced. After the brief introduction students were given a case study on workflow-intensive Web application. This was used to identify any changes to their understanding to the introduced aspects and to identify the initial understanding of specific aspects required to model workflow intensive Web applications. Table 6.9 gives the analysis of the responses to questionnaire 2. The answers were analysed using the same method used in analysis of the background study.

Table 6.9: Questionnaire 2-Students responses to the conceptual model and aspects of Workflow intensive Web applications

<table>
<thead>
<tr>
<th>Aspect</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No</td>
</tr>
<tr>
<td>Conceptual Model</td>
<td>6.67%</td>
</tr>
<tr>
<td>Presentation</td>
<td>40.00%</td>
</tr>
<tr>
<td>Composition</td>
<td>33.33%</td>
</tr>
<tr>
<td>Data</td>
<td>20%</td>
</tr>
<tr>
<td>Navigation</td>
<td>20%</td>
</tr>
<tr>
<td>Access Control</td>
<td>40%</td>
</tr>
</tbody>
</table>
According to Table 6.9, 87% gave a close or matching definition of the conceptual model. More than 50% closely or perfectly identified aspects such as composition, data, navigation and access control. 27% identified the workflow aspect while 13% gave a close answer to that.

6.4.3.2 Assignment 1

Assignment 1 was on developing an information centric Web application. From the background study, it was revealed that 86.67% of students are familiar with HTML and other technologies required in developing information centric Web applications. Students were provided with the CBEADS framework with CMS application to be used in the assignment to allow students to develop Web applications without making technology a barrier. However, they were given the opportunity to select any other technologies according to their preferences.

Students presented their Web applications and the development process of Web applications to the class. The Web sites produced by the students were evaluated in terms of usage of aspects required to develop an information centric Web application. The usage of the aspects in their assignments is given in Table 6.10. One Web site couldn’t be evaluated because of a technical problem. One student had only done the navigation and template for his Web site.

The students’ choices of technology were surveyed along with the reasons for their selection. 80% of students had different technologies other than the CBEADS framework for the development of the Web site. The reason given by students for their

<table>
<thead>
<tr>
<th>Workflow</th>
<th>33.33%</th>
<th>26.67%</th>
<th>13.33%</th>
<th>26.67%</th>
</tr>
</thead>
</table>

choice of the technology was “the known technologies were sufficient for the assignment tasks”. Therefore, they weren’t willing to try new technology for the assignment. The students’ behaviour in this situation is consistent with Rosson et al.’s finding on end-user behaviour. They explained that end-users attempt to use the known tool for developing all the different applications they want (Rosson, Ballin, & Heather, 2004). One of the two students who was not familiar with HTML has used CBEADS for his Web site. The other student learnt HTML and frontpage for his Web site creation. According to him, the reason for his choice was that he wanted to learn HTML and CSS.

Table 6.10: Assignment 1- Usage of the aspects for the development of the Web sites

<table>
<thead>
<tr>
<th>Aspect</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Presentation</td>
<td>86.67%</td>
</tr>
<tr>
<td>Content</td>
<td>80%</td>
</tr>
<tr>
<td>Navigation</td>
<td>86.67%</td>
</tr>
</tbody>
</table>

6.4.3.3 Week 9 Tutorial

The week 9 tutorial was involved in modelling the workflow of a workflow intensive Web application using aspects at the conceptual level. Students were requested to send the answers to the lecturer in charge. Only six students sent the answers. The analysis of the models students provided shows that the students had specified the attributes of the aspects sufficiently to model the workflow.
6.4.4 Evaluation: 4th step of the AR cycle

With the formal introduction of the conceptual model, close or matching answers for the definition of the conceptual model increased by 40%. The conflicts in understanding appeared to have reduced by 33.37%. This was a major improvement. Even though the understanding of aspects after the lecture improved, this doesn’t show a significant level of difference to the same degree as response to the conceptual model. However, the assignment tasks show a significant improvement in the students’ understanding of the aspects. The assignments provide students with an opportunity to practice modelling the conceptual level of Web applications.

6.4.5 Specific Learning: 5th step of the AR cycle

The initial findings with the students identify that the students do not possess a clear understanding of the modelling of Web applications. Even though they had developed Web applications for their assignments, they could not identify all the aspects required to model Web applications at the conceptual level. It is suggested that providing case studies to practise in the class can help students to understand the aspects required to model different types of Web applications. A meta-model based model driven Web application development framework can be used to reinforce the modelling aspects by allowing them to realise the models they produced. Students can change the specifications of the models and observe the changes at runtime to the Web applications. This will further help the students to understand the mapping from the conceptual domain to the physical Web application. The strategies presented in Table 6.11 to help students to learn modelling Web applications at the conceptual level are formulated based on the observations.
Table 6.11: Strategies to be used in teaching modelling

<table>
<thead>
<tr>
<th>Factor</th>
<th>Strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Practical Experience</td>
<td>The meta-model framework should be provided to the students to create the applications from their models. It will help them to check the completeness of the models they have created.</td>
</tr>
<tr>
<td>Infrastructure</td>
<td>The infrastructure should be provided to the students thus the class can be fully focused on the modelling.</td>
</tr>
<tr>
<td>Guidance</td>
<td>Guidance should be provided in the case studies and tutorials to understand the good and bad models.</td>
</tr>
</tbody>
</table>

6.5 Strategies to be used with Meta-design paradigm

The reflection on the experience collected from the three studies discussed above, leads to the establishment of strategies to be used with the meta-design paradigm. These strategies are presented in Table 6.12.

Table 6.12: Strategies for End-user development

<table>
<thead>
<tr>
<th>Factor</th>
<th>Strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Characteristics of Individual</td>
<td>The individuals should be self motivated and keen to learn new technologies.</td>
</tr>
<tr>
<td>Support</td>
<td>More support should be provided during the development phase to get the application developed according to the</td>
</tr>
</tbody>
</table>
requirements. Then end-users are able to manage most of
the activities during the maintenance phase.
As identified in the action research cycle, it would be
beneficial to give the end-users an opportunity to self judge
the good and bad practices in modelling the applications.

<table>
<thead>
<tr>
<th>Infrastructure</th>
<th>Management of infrastructure should be outsourced to reduce the barrier to end-user development.</th>
</tr>
</thead>
</table>

### 6.6 Conclusion

The objective of the studies was to demonstrate how the aspects of the meta-design paradigm based on meta-model, infrastructure, learning environment, and socio-technical environment, can support end-user development. The first study shows how the meta-model based Web development infrastructure can support evolution of the Web applications. The categorisation of the complexity of the tasks helps to identify the activities that end-users can participate in. This helps to balance between the DIY approach and the professional developers in the development activities. The meta-model based meta-design paradigm can help end-users to gradually become end-user developers. The last study demonstrates the application of the meta-design paradigm in class room to facilitate the students to learn to model the Web applications. This also shows how the aspects of Web applications can be used to teach modelling of Web applications at the conceptual level. Even though the findings in chapters 3 and 4 reveal that conceptual model is closer to the end-user’s mental model, the awareness of the aspects of the conceptual model can significantly increase the opportunity of creating quality Web applications. The studies also helped to establish the strategies to be used to support end-user developers: 1) Self motivated individuals, who are keen to use new
technologies; 2) Provide more support during the development phase; and 3) Outsource the management of the infrastructure. These studies demonstrate the practical functioning of the meta-model based meta-design paradigm. This completes the answer to “How to support business end-users to develop / and maintain Web applications to support their business processes”.
7 Conclusion and Future Work

7.1 Introduction

This thesis addressed a significant issue within the EUD Web domain – the absence of an end-user development approach, designed specifically for business users, particularly SMEs. The requirements of a suitable end-user development approach were established based on the literature review, study with SMEs, and a study on end-user development by business users. A meta-model based meta-design paradigm is proposed in this thesis as a possible solution to address the requirements of a suitable end-user development approach for SMEs. A meta-model of Web applications is developed based on patterns of different types of Web applications required by SMEs. A meta-model can be instantiated by specifying the values for the attributes of the meta-model to create specific Web applications. Also Web applications can be evolved by changing the values for the attributes of the meta-model. The attributes of the meta-model are derived from the attributes of the conceptual model of Web applications. End-user development concepts and activities discussed in chapter 2 have been used in the implementation of the activities to specify the attributes of the meta-model. The activities to specify the attributes of the meta-model are grouped in to different complexity levels. The properties of the meta-model and the categorisation of the complexity levels help end-users and developers to collaborate in the development and maintenance of Web applications. The meta-model based meta-design paradigm is practically evaluated with a group of users including SMEs and students. These studies help to establish strategies to be used with the meta-model based meta-design paradigm.
This chapter is organised as follows. First, it summarises the research presented in this thesis. Secondly, the contributions made in the course of this research project are presented. Next the limitations of the research are discussed. Finally, the chapter concludes by highlighting various possible future research directions.

7.2 Research Summary

The overall aim of the thesis was to provide an answer to the research question: “How to support SME end-users to develop and / or maintain the Web applications to support their business processes?” Figure 7.1 summarises the research approach.

![Research Flowchart]

**Figure 7.1: Research Summary**

Related work on end-user development, Web engineering and psychology of programming were explored to identify a suitable development approach for SMEs. This literature survey identified a gap namely the need for a development approach that can support development
of different types of Web applications required by SMEs. The related work is presented in chapter 2. The practical relevance of the research question was investigated by studying the social and managerial aspect of end-user development in the context of SMEs. This study also helps to identify the capabilities of SMEs in relation to the development of Web applications. End-user development in a large project within an organisation was studied to identify the issues and benefits of end-user participation during the design and development of Web applications. These two studies detailed in chapter 3 together with the findings from the literature survey established the following requirements for an end-user development approach:

1) Need to support different types of Web applications required by SMEs;
2) Need to support the specification of Web applications at the conceptual level;
3) Need for a common data repository to store the data used in different applications within the organisation;
4) Common login to all applications within the organisation;
5) Balance between Do it Yourself (DIY) and professional developer that allow end-users to perform the activities they are capable of while getting the help from a professional developer to do the difficult tasks.

The Web applications required by SMEs can broadly be categorised into two groups namely information centric Web applications and process centric Web applications. Information centric Web applications are simple Web sites with unstructured information where the focus is on effective presentation of information. Process centric Web applications are Web sites that support business processes by enabling users to perform actions such as filling a form, and approving a form. Process centric Web applications can be further grouped into
data intensive Web applications and workflow intensive Web applications. The aspects required to model these different types of Web applications are given in Table 7.1. Chapter 4 detailed the aspects required to model these different types of Web applications at the conceptual level.

Table 7.1: Aspects required to specify different types of Web applications

<table>
<thead>
<tr>
<th>Type of Web applications</th>
<th>Aspects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Information Centric</td>
<td>Presentation, Navigation, Content, Access Control, Personalisation</td>
</tr>
<tr>
<td>Data intensive</td>
<td>Presentation, Navigation, Data, Access Control, Personalisation, Task</td>
</tr>
<tr>
<td>Workflow Intensive</td>
<td>Presentation, Navigation, Data, Access Control, Personalisation, Workflow</td>
</tr>
</tbody>
</table>

The usage of these aspects in specifications written by end-users was analysed. This study revealed that end-users focused only on specific aspects that they believe are important in their applications such as data and process. This finding also leads to a requirement for the development approach:-; Need to support creating Web applications with minimum aspects.

The meta-design paradigm based on a meta-model is constructed to support the identified requirements. The meta-model is organised into three levels: shell level, application level, and function level. The aspects common to all Web applications such as navigation and access control are modelled at shell level. The aspects common to specific Web applications such as workflows are modelled at application level. The requirements specific to a use case within an application are modelled at the function level. The activities to create the Web
applications based on the meta-model were grouped into three different complexity levels: routine level, logic level and programming level to tackle the problem of divergent skill levels among the users, or affordability for learning complex tasks as suggested by different researchers such as Beringer (2004) and Burton et al. (1984). The CBEADS framework was extended to support the meta-model based development of different types of Web applications. After creating the application, end-users can modify the Web application by changing the attribute values of the aspects. The end-user development concepts discussed in chapter 2 were used in creating the meta-tools to instantiate and modify the Web applications. Chapter 5 presents the meta-model based meta-design paradigm.

The qualitative studies outlined in chapter 6 evaluate the meta-design paradigm. The first case study was conducted with a group of SMEs. Surveys and lab experiment were used in this research. The second study was conducted with a group of students who volunteered to complete an assignment which consisted of activities to develop Web applications using the CBEADS framework. The third study was conducted as an action research cycle which investigated the way students used the aspects in their specifications. A set of strategies for the success of the meta-design paradigm based on the meta-model were established based on the findings from these studies in these three areas: individual characteristics of the end-users, facilitation, and infrastructure.

7.3 Contributions of this Research

The original contributions of this thesis enhance the end-user development discipline by widening the benefits of end-user development to business end-users, particularly to SMEs.
More importantly the major contributions of this research also provide a practical tool that can be used by end-users with little or no previous experience in Web application development to develop the Web applications that they require to support their business processes. The findings can be adopted in end-user development tools to better support the needs of the SMEs. The major contributions of the thesis are listed below.

1. A set of requirements for a suitable end-user development approach that can be used by Small to Medium Enterprises. The requirements were derived from the literature survey, studies with a group of SMEs and a study on a user involved in end-user development and an analysis of specifications for applications written by users.

2. The sets of aspects required to create conceptual models for different types of Web applications. The sets of aspects are derived from the literature survey on existing conceptual models of different types of Web applications. Then the attributes of these aspects that are required for specifying the Web applications are refined through modelling example applications.

3. A meta-design paradigm based on the meta-model of Web applications to support DIY approach to Web application development. The meta-model is based on the patterns of different types of Web applications. The meta-model based development approach creates the infrastructure to develop evolvable Web applications. A meta-model based environment with meta-tools provides a learning environment that can suit end-users with different levels of skills in Web application development.
4. Strategies for successfully using the end-user development approach were formulated in the qualitative studies conducted with the group of SMEs and students. These can be used with end-user development environments and tools particularly, focused on SMEs.

7.4 Limitations

Although this work has proven to be useful from both the practical and theoretical points of view, some limitations are acknowledged. The primary cause for these limitations is the time constraints, which forced certain milestones such as completion of thesis writing, to be achieved by deadlines. However, other identified limitations of this research are presented below to indicate possible continuous improvement of this work.

Currently the meta-model based meta-design paradigm is predominately a theoretical construct with limited technology support. The CBEADS framework was used to instantiate the meta-model but there’s currently limited facility to automatically generate the process intensive Web applications within the framework. The full potential of the meta-model based meta-design paradigm will not be realised without further attention to the CBEADS framework.
The study on the managerial and social perspective of SMEs was conducted with a sample of SMEs mainly in the tool making industry. As access to all SMEs in Australia was not possible, generalising the findings to all SMEs constitutes a limitation of this research.

As mentioned before, the meta-design paradigm for development was evaluated with a group of students in University of Western Sydney. This was due to time and financial constraints. Although this evaluation helps to evaluate the success of the meta-design paradigm with a group of users who have a similar mental model of Web application development, this approach does not provide an opportunity to evaluate the construct with SMEs. Future case studies or action research cycles with SMEs could address this limitation.

7.5 Future Work

The contributions of this thesis set the scene for additional research work to further the domain of end-user development in the following areas. Some of the major improvements and research directions are explained in this section.

- The meta-model can be supported by a model driven development framework such as CBEADS. CBEADS can instantiate data intensive Web applications and information centric Web applications at logical and routine levels. However, the two engines: WFE (workflow engine) and VTMAC (Virtual team based access control) need to be extended to support creation of workflow intensive Web applications at the logical level.
• Additional engines within the CBEADS framework are required for instantiation of Web applications with personalisation aspects.

• In this study mainly form based tools were used to specify the aspects such as workflow and forms. Further research is needed into development of better methods of capturing the inputs to instantiate the meta-model.

• Additional testing of the meta-design paradigm with SMEs is required to claim the full benefits of the proposed approach. Action Research cycles can be used to look at the success of the evolution of the Web application within SME organizations.

7.6 Conclusion

This chapter summarised the main findings and implications of the research. The research work illustrated in this thesis confirms that end-user development of Web applications can be effectively achieved, by a meta-design paradigm based on the meta-model of Web applications. Finally, this chapter presented the limitations of the research and possible future research directions. The outcome of this research can be viewed as the beginning of a number of new research studies.
References


*RDF Primer XML 1.1.* (2004).): World Wide Web Consortium W3C.


Appendix A- CBEADS Framework

Component Based E-Application Deployment / Development System (CBEADS) was initially created in 1998 with basic functionality of user authentication, role based access control, ability to add more functions and assigning these newly created functions to users. The ability of adding more functions to CBEADS helps framework to support incremental development. CBEADS framework consists of two main components called system and applications. System components can be further grouped in to three component types called; (a) tools, (b) engines, and (c) the shell. Tools components can use to create new functions using CBEADS framework. Engine components help to execute the functions. Applications are collection of functions. CBEADS also can use to develop application to be deployed outside the framework. For example a static web site generated using ‘content management system’ tools could be deployed with an Internet Service Provider (ISP). The shell contains modules that are common to tools, engines and applications to operate smoothly. CBEADS architecture is shown in Figure A.1. CBEADS environment as a whole depend on basic web infrastructure such as operating systems, and web servers for its operation.

When adding new components to CBEADS developers need to decide whether that needs to be a tool, engine or a module in CBEADS shell and register them accordingly. Tools are the components that are needed for creating and assembling ‘component parts’ of an application, for example the front end pages, the business rules, back end databases, etc. Engines are the components that support the runtime operation of the application. When a
certain module is needed by tools, engines or applications it is implemented as a module in the core shell. For example functionalities such as user management, access control, session management, etc. are needed by tools, engines and applications. Hence they are in CBEADS core shell. CBEADS tools and engines are summarised in Table A.1 and A.2. These tools and engines are explained in from Appendix B to E.

Figure A.1: CBEADS framework; Tools, Engines, Applications and Shell
### Table A.1: Tools in CBEADS

<table>
<thead>
<tr>
<th>Tools</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smart Business Object (SBO Builder)</td>
<td>Use to specify and create the data model in a process centric application.</td>
</tr>
<tr>
<td>Interface Renderer Tool (Renderar)</td>
<td>Use to specify and generate the interfaces based on data model created by SBO builder.</td>
</tr>
<tr>
<td>Workflow Management System (WMS)</td>
<td>Use to specify the workflow in a process centric Web application.</td>
</tr>
<tr>
<td>Content Management System (CMS)</td>
<td>Use to specify and generate static Web sites.</td>
</tr>
</tbody>
</table>

### Table A.2: Tools in CBEADS

<table>
<thead>
<tr>
<th>Engine</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workflow Engine</td>
<td>Use to execute the state based workflow specified by WMS.</td>
</tr>
<tr>
<td>Virtual Team based Access control</td>
<td>Use to resolve the context of artefacts at runtime when managing the workflow. E.g.: immediate supervisor of given person.</td>
</tr>
</tbody>
</table>
Appendix B: SBO Builder and Renderar

SBO builder in CBEADS can be used to specify and generate business objects for the applications with in CBEADS. Business objects are specified using SBOML language. SBOML attributes support convention settings such as file upload facility for documents, displaying URLs as hypertext links, rendering a calendar to assist user to enter date information. This enables Renderar to auto generate appropriate web interfaces based on business objects. For example, in Figure B.1 employee business is rendered as a Web table with search capability. In this example, the contents of the email address attribute are displayed as mailto hypertext links and the content of the photo attribute is displayed as images.

Figure B.1: Rendering SBO as a table
SBO consists of a default schema to control the global behaviours of all business objects. The schema defines a set of default templates used by each rendering API. Thus, different views can be generated on the business objects by using different templates in the default schema or specifying custom schemas. The schemas also define the behaviour of attribute types such as validations, localisation, option values, and formatting and conversion of values. The renderer generates the appropriate Web user interfaces for the business objects with the selected attributes based on the types of the attribute. SBO also use types for the attributes based on their nominated attribute type given when the SBO was modelled. If the attribute type is not explicitly defined when specifying the business object, SBO builder will logically derive the most suitable (conventional) web user interfaces for those SBO attributes at run-time. For example, if it is specified an attribute for employee as “employee has email”, then SBO automatically use the type as e-mail for that attribute without extra declaration.

**SBOML**

SBOML allows users to express their domain specific business objects in near natural language syntax. It has a set of predefined high level attributes types: description, answer, reminder, text, name, alphanumeric, date, number (float), integer, id, email, first name, last name, time, money, cost, price, balance, total amount, password, phone, mobile, file, document, image, photo, public document, public image and url.

SBOML syntax is given below:

Following conventions has used in specifying the SBOML syntax:

- Keyword elements are emphasised in both bold and italic.
- Normal style texts represent user-defined elements.
• When an element consists of a number of alternatives, the alternatives are separated by a vertical bar (“|”).
• Optional elements are indicated by square brackets (“[” and “]”).
• An ellipsis (“...”) indicates the omission of a section of a statement, typically refers to recursive statements.

in namespace, business object has attribute A [(mandatory] [type] [which could be option a or | and option b]],[might have] [many] another business object [(has attribute B, attribute C, yet another business object (has ...)]...,[ use method A (method name type from location [option is value,...] [with attribute A, attribute B,... [with attribute A as parameter name abc , attribute B as parameter name ...]), service B...]

in namespace, business object has | might have another business object [as attribute X] [via yet another business object]

The “in” clause defines the namespace where the subsequent business object(s) are created within. If the namespace does not exist, a new namespace is created. The “has” clause defines the attributes of the intended SBO, or ‘has’ relationships with another intended SBO. The optional “use” clause defines the methods (operations) of the SBO. The “as” clause is used to refer to an attribute of a foreign SBO instance. The “via” clause is a shorthand for specifying “many-to-many” relationships. Business objects and relationships can be specified using the editor as shown in Figure B.2.
Once the business object is defined, UI Renderar allows to specify the interface on the business object, then it generates new interface based on the specification. The create UI function is shown in Figure B.3. This function allows defining different UI types such as tables, menus, forms. Table option generates a report view on the selected business object. Form option generates a form view on the selected business object. Menu option gives a tabbed form with a tab for each object and all related objects.
Figure B.3: Interface Renderar
Appendix C: Content Management System (CMS)

Content Management System (CMS) allows creating and maintaining a static web site. CMS provide following functions:

- Create a Static Web Site
- Manage the Site Structure
- Manage Information
- Manage Templates
- Publish Site

These functions can be accessed in CBEADS menu under CMS as shown in Figure C.1.

Figure C.1: CMS functions
New Web site can be created using “create site” function in CMS application. It will create the folder structure to store the pages and add a record to the list of Web sites managed in this site. It will use the default template for the Websites. Also it will create the blank home page for the Website. Then it will take the user to “Website Information Structure” page where he/she can create and edit pages of the Web site.

![Create New Web Site](image)

**Figure C.2: Create Site Page**

The “Website Information Structure” page shows the site structure with links to “Add Page”, “Edit Page” and “Delete Page” as given in Figure C.3.
Users can add pages to the Web site by using “add page” function. Valid page name can contain only alphanumeric characters. CMS tool will create a HTML page with the selected template for the Web site. Then the user can add the content to the Web site using edit page function.
Content of a page can be created using “edit page” function as shown in Figure C.5. “Edit page” function consists of a WYSIWYG Editor. It can support adding and editing text and images to the pages. The user can preview the created page.

![Edit Page](image)

**Figure C.5: Edit Page**

The menu of the Website can be managed using “Menu Management” function. You can define the order of the pages and the display name for each menu item as shown in figure C.6.
Web sites created through CMS can be uploaded to ISP using “publish site” function as shown in Figure C.7. The access data (FTP address, user name and password) of ISP is saved to text file when it is entered first time. After that this data is used to make the FTP connection.
Appendix D: Workflow Management System (WMS)

Workflow Management System (WMS) stores the specification of state based workflow within an existing application into a database. Users have to specify the name for the workflow as shown in Figure D.1.

Figure D.1: Create Workflow
Users can specify the states of a workflow as shown in Figure D.2 state has an entry condition which is to be tested when entering to that state and a position which shows the current step of the workflow which is executing. A state has many actions. Action consists of an actor, User interface, System action, Exit condition, next state, options.

![Manage State](image)

**Figure D.2: Manage State**

Option of a state specifies an alternative execution criterion in a state. Therefore, each option is mapped to a button in the user interface for the state. For each option, user has to specify the name, exit condition, system action and next state as shown in Figure D.3.
Figure D.3: create Option

Workflow Engine executes the workflow specification. It generates the notifications and manages the access to workflow. The engine understand number of built-in actions such as “go to state” for changing states and querying states. It also supports any number of additional actions given to it that it can return at appropriate times back to its calling application which is then responsible for their execution.
Appendix E: Virtual Team Based Access Control (VTMAC)

Virtual Team based Access Control (VTMAC) manages the project team in an application. It will create a virtual team to participate in the workflow. It controls the access to the different resources within a project. For example, in course approval system, if an academic from School of Computing and Mathematics submit a course proposal it needs to be routed to Head of School of that school. The virtual team for school of computing and mathematics can be specified using “create project team” interface as shown in Figure E.1. Project team can be changed during the runtime of the application through this interface. The changes are applied to all new workflow instances that executed after that.

![Figure E.1: Create Project Team Interface](image-url)
Appendix F: SME Questionnaire 1

CONFIDENTIAL

Hands on Workshop Questionnaire

Age Group: □ 18-30 □ 31-45 □ 46-60 □ Over 60
(Please tick the relevant box)

1. What is your IT Background? (Workshops you’ve attended, Courses you’ve followed)

________________________________________________________________________

2. Have you used any application programs? (Please tick the relevant box)
   □ Microsoft Word □ Spreadsheet
   □ e-mail □ Account Keeping Software
   □ Other (Please Specify)
________________________________________________________________________

3. Have you used any Programming Languages and Databases?
   □ Visual Basic □ Access
   □ Other (Please Specify)
________________________________________________________________________

4. Do you have any experience related to web site development?
   - Have you done Information Design for Web site? □ Yes □ No
   - Have you done any Web site development before? □ Yes □ No

If you have done any web site development yourself, please rate your experience.
Ratings: 0: No
1: I told what I want to a Designer and he did the job for me
2: I did with some one else. (Active Participation)
3: I did it myself

e.g.: VB 0 1 2 3

What are the Programming languages you used? (Please rate the experience)

VB 0 1 2 3
(other, specify) 0 1 2 3
(other, specify) 0 1 2 3

© AeIMS Research Group-UWS
• Have you used any software packages? (Please rate the experience)

Dreamviewer 0 1 2 3
FrontPage 0 1 2 3
(和其他) 0 1 2 3
(其他, specify) 0 1 2 3

• Have you done any website maintenance? ☐ Yes ☐ No

If yes, please describe what you did. (eg. I have updated the product information)

If yes, what tools did you use?

4. What interest do you have in DIY (Do-it-yourself) approach to website development?
Please tick the relevant boxes given below.

☐ Hobby
☐ Minimise Cost
☐ Ease of Maintenance
☐ Other (Please Specify)

5. What are the problems you see in adopting a DIY approach to website development?
Please tick the relevant boxes given below.

☐ Doesn’t know a tool
☐ Doesn’t have time to learn
☐ Doesn’t have time to do
☐ Believe in conventional development
☐ Other (Please Specify)

6. What do you expect from this ‘Hands on’ Workshop?

© AdIMS Research Group-UWS
Appendix G: SME EUD workshop

Notes

DIY Approach to WebSite Development and Maintenance: Hands on Workshop
Application Development Team
University of Western Sydney

Activity 1

• Fill the Questionnaire on related experience

Outline

• Why DIY approach?
• How to do it?
• Website Development
• Website Maintenance

Why DIY?

• Options to develop websites
  • Contracting out
    • Get software company/person to develop the website.
  • Do it yourself (DIY)
    • Develop the website on your own.

Pros and Cons

<table>
<thead>
<tr>
<th></th>
<th>Contracting out</th>
<th>DIY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development Cost</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Maintenance Cost</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Time</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>Require Knowledge</td>
<td>Low</td>
<td>High</td>
</tr>
</tbody>
</table>

Where to Start?

• What is the purpose of the website?
• How will you organize the appearance of the site information (site structure)?
• How will you design the look and feel for the site?
• How will you publish a site on the Web?
• How will you maintain a site?
Activity 2
- Plan your website
  - What is the purpose of the website
  - What information you want to have on site
  - Draw the site structure

Activity 3: How to do it?
- Getting Started
  - Log in to Computer
  - Log in to CBEADS
  - Select CMS (Content Management system)
- First Step
  - Create a site

Activity 4: How to do it?
- Organizing Information
  - Add Page (similar to create a doc file)
  - Delete Page (similar to delete a doc file)
  - Edit Page
    - Text
    - Image

Activity 5: How to do it?
- Site Structure
  - Menu Creation

Activity 6 & 7: How to do it?
- Last Step
  - Publish the site
- Change a page and republish, (optional)

Discussion
Appendix H: SME EUD

Questionnaire 2

CONFIDENTIAL

Hands on Workshop Feedback Form

(i) What new knowledge have you gained?

____________________________________________________________________

____________________________________________________________________

(ii) How difficult was it to do the tasks? (Please circle the correct number)

1: I can do it on my own by looking at the manual
2: I can do it after explanation and referring to the manual
3: I need some assistance at first time, but can do it on my own after
4: I can’t do it, it’s too difficult.

Site Creation:  1  2  3  4
Organizing Information:  1  2  3  4
Doing the Site Structure:  1  2  3  4
Publish the Site:  1  2  3  4

(iii) What features/enhancements do you think are required in the CMS for you to do it easily and better?

____________________________________________________________________

____________________________________________________________________

(iv) Do you think you’d be interested in working on site development using the CMS tool after the workshop? □ Yes □ No

If yes, what kind of assistance do you think you require?

____________________________________________________________________

(v) Would you like to make any other comments?

____________________________________________________________________

____________________________________________________________________

© AeIMS Research Group-UWS
Appendix I: End-user Developer Interview Questionnaire

1. What are the tasks you have completed during the OCAS design and development time?

<table>
<thead>
<tr>
<th>No</th>
<th>Task Description</th>
<th>Facilitating features</th>
<th>Difficulties</th>
<th>Any other features to facilitate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

249
2. Any other comments

Name:

IT Qualifications:

Job Description:

Role in Online Course Approval System:
What is the job relationship to Course Approval System:

Experience in using computer (please X the box if you have experience with each application)

<table>
<thead>
<tr>
<th>Applications</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>[ ] Microsoft Word</td>
<td>[ ] Spreadsheet</td>
</tr>
<tr>
<td>[ ] e-mail</td>
<td>[ ] Account Keeping Software</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Programming Languages and Databases</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>[ ] VB</td>
<td>[ ] Access</td>
</tr>
</tbody>
</table>


Appendix J: EUD-Questionnaire 1

<table>
<thead>
<tr>
<th>Name</th>
<th>.........................................................................................</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student ID</td>
<td>.........................................................................................</td>
</tr>
<tr>
<td>Project Group</td>
<td>.........................................................................................</td>
</tr>
<tr>
<td>Project Role</td>
<td>.........................................................................................</td>
</tr>
</tbody>
</table>

1. Circle the relevant number according to your expertise in web application development (1-no knowledge, 5-expert knowledge).

1. How do you rate your knowledge of web application development?
   1  2  3  4  5

2. How do you rate your knowledge of HTML?
   1  2  3  4  5

3. How do you rate your knowledge of cascading style sheets?
   1  2  3  4  5

4. How do you rate your knowledge of client-side scripting (java script, etc.)?
   1  2  3  4  5
5. How do you rate your knowledge of server side scripting (PHP, ASP, JSP, Perl, etc.)?
   1  2  3  4  5

6. How do you rate you experience of SQL?
   1  2  3  4  5

2. Rate the following specific concerns in web application development according to your capabilities (0-don’t know, 1-extremely difficult, 7-very easy). Also please specify the technologies you have experienced in the provided space.

1. Availability and configuration of server software
   0  1  2  3  4  5  6  7

2. Configuration of development environment
   0  1  2  3  4  5  6  7
3. Design and develop of user interface layout

0  1  2  3  4  5  6  7

4. Designing graphics

0  1  2  3  4  5  6  7

5. Relational Database design

0  1  2  3  4  5  6  7

6. Object Modeling

0  1  2  3  4  5  6  7
7. Authentication and authorisation

0  1  2  3  4  5  6  7

8. Ensuring security (encryption, protocols, etc.)

0  1  2  3  4  5  6  7

9. Debugging

0  1  2  3  4  5  6  7

10. HTTP protocol and session management

0  1  2  3  4  5  6  7
11. User input validation

\[
\begin{array}{cccccccc}
0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
\end{array}
\]

12. Create databases

\[
\begin{array}{cccccccc}
0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
\end{array}
\]

13. Storing form data in database

\[
\begin{array}{cccccccc}
0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
\end{array}
\]

14. Reading data from a database table and display on a web table

\[
\begin{array}{cccccccc}
0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
\end{array}
\]

256
15. Normalising databases

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

16. Indexing tables

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

17. Implementing business rules

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

18. Ensuring usability

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>
3.1 Do you have experience in web application development?

3.2 How long have you been developing web applications?

3.3 Specify few of the applications you have developed?

4. List Information Technology subjects you have followed before?
Appendix K: EUD Assignment 1

End-user Development – Workshop


2) Login to CBEADS in “http://web2.scm.uws.edu.au” server on the firefox browser. User name: student ID

password: first four digits of student ID.

3) In today session we go to develop an application with two use cases for the following case study.

University organizes series of workshops to help students to improve the skills in relevant areas such as communication, writing, etc. The figure 1 gives a partial class diagram for the workshop registration system.
i) Create the objects using SBO for the following class diagram. Use your student ID as the namespace Refer to the manual for the syntax for SBOs.

ii) Create an application in CBEADS with your student id.
Application Description: Test Application

iii) Add a function called “Add workshop” to your application.
Description of the function: Manage workshops

iv) Assign the newly created function to the user group with your project group name. (This is the project group you worked in for project subject)

v) Create the menu given in figure 2 to manage workshop, using the namespace “wrs” using the interface renderar. Select the ‘create’ option for general options.
vi) Add a function called “View Workshop” to the application you created in section 2. Function Description: View the existing workshops.

vii) Create the report given in figure 3 to display workshop data, using the namespace “workshop” using the interface renderar.
Please use the Explora browser for following activities.

viii) Create a web site using CMS for the Workshop Registration System. Use your student ID as the site name.
ix) Add page named “WRS” to your site.
x) Add the content to make the “WRS” page looks as follows.
(You can use the images given at “download image” option under CMS in CBEADS menu. Upload the image to your website and insert it to the page.)
Appendix L: EUD Questionnaire 2

<table>
<thead>
<tr>
<th>Name</th>
<th>.........................................................................................</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student ID</td>
<td>.........................................................................................</td>
</tr>
</tbody>
</table>

Rate the following aspects in using Component based Development and Deployment framework (0-didn't try, 1-extremely difficult, 7-very easy).

1. Login to CBEADS
   
   0 1 2 3 4 5 6 7

2. Accessing SBO and interface renderar
   
   0 1 2 3 4 5 6 7

3. Using tools for developers
   
   0 1 2 3 4 5 6 7

4. Creating object definitions and relationships using SBOs
   
   0 1 2 3 4 5 6 7

5. Creating form with interface renderar
   
   0 1 2 3 4 5 6 7
6. Displaying data with interface renderar

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
</table>

7. Creating new application

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
</table>

8. Adding new function

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
</table>

9. Assigning functions to the groups

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
</table>

10. Creating website

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
</table>

11. Creating a page for the website

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
</table>

12. Uploading images

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
Appendix M: End-User

Specifications-Questionnaire 1

End-user Development Questionnaire 1

<table>
<thead>
<tr>
<th>Name</th>
<th>...........................................................................................................</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student ID</td>
<td>...........................................................................................................</td>
</tr>
</tbody>
</table>

1. Answer the following questions.

1.1 Do you have experience in web application development?

1.2 How long have you been developing web applications?

1.3 Specify few of the applications you have developed?
1.4 What are the technologies you are familiar with?

1.5 What do you mean by a conceptual model of a web application?
2. You have to specify different aspects to develop a product catalogue given below. Assume there is a requirement for a web interface to update the product catalogue by the Sales Clark. Could you please describe the different aspects you have to specify when developing a web site like this? Your description may include the different elements of the aspects. Also describe how these different aspects can be related to one another.

Aspect 1:
Aspect 2:

Aspect 3:
Aspect 4:

Aspect 5:
Aspect 6:
Appendix N: End-user Specification

Questionnaire 2

End-user Development Questionnaire 2

<table>
<thead>
<tr>
<th>Name</th>
<th>...........................................................................................................</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student ID</td>
<td>...........................................................................................................</td>
</tr>
</tbody>
</table>

1. What do you mean by a conceptual model of a web application?
2. Assume you have to analyse the requirements for leave processing system for a Company. According to the company policy, there are few different leave types such as annual, sick and no-pay. Once a staff member applied for annual leave the leave application should be approved by the Head of Department and then forward to the HR office for recording. However, if the staff member applied for no pay leave, then it should approved by the Head of Department and then by the General Manager. If any staff member applied for sick leave it will directly go to HR office. Leave processing system needs to be developed as an intranet application.

Could you please name and describe the different aspects that you think is required to specify when developing a web application like this? Your description may include different elements of the aspects. Also describe how these different aspects can be related to one another.

Aspect 1:
Aspect 2:

Aspect 3:

Aspect 4:
Aspect 5:
Appendix O: Assignment 1-

Feedback form

Student Feedback Form 1-CMS

<table>
<thead>
<tr>
<th>Name</th>
<th>.................................................................</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student ID</td>
<td>..............................................................................</td>
</tr>
</tbody>
</table>

Did you use CBEADS in assignment 1? Please circle the correct answer. Yes/No

If you circle yes for the question 1, please go to part A in the questionnaire. Otherwise go to part B in the questionnaire.

Part A

1) Why did you select CBEADS for the assignment? Please circle most appropriate answer(s). Use the space to write down any other reasons for your selection.

<table>
<thead>
<tr>
<th>i.</th>
<th>I don’t know any other CMS Application.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ii.</td>
<td>CMS had better features than the tool I known before.</td>
</tr>
<tr>
<td>iii.</td>
<td>I’m not familiar with HTML.</td>
</tr>
</tbody>
</table>

2) If you select ii for question 1) please brief the comparison.
Please rate your sentiment towards the statements based on your experience with CMS.
Place circle the option which best describes your opinion. If you are not agreeing with the statements please briefly explain the limitations of the CMS which give you that sentiment.

3) CMS allows specifying the presentation aspects as I wish.
   1) Agree Strongly    2) Agree    3) Indifferent    4) Disagree    5) Strongly Disagree

4) CMS allows creating the structure of the site as I wish.
   1) Agree Strongly    2) Agree    3) Indifferent    4) Disagree    5) Strongly Disagree
5) CMS allows creating the content of the site as I wish.
   1) Agree Strongly  2) Agree  3) Indifferent  4) Disagree  5) Strongly Disagree

6) CMS allows creating the navigation structure of the site as I wish.
   1) Agree Strongly  2) Agree  3) Indifferent  4) Disagree  5) Strongly Disagree

7) Please use this space to write any usability issues you had with CMS.

Part B

8) What are the development tool/editor/CMS you use in the assignment?
9) Why did you go for that selection in the assignment? Please circle appropriate answer(s). Use the space to write down any other reasons for your selection.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>i.</td>
<td>I didn’t have time to learn another Application.</td>
</tr>
<tr>
<td>ii.</td>
<td>The tool I used has better features than CBEADS.</td>
</tr>
<tr>
<td>iii.</td>
<td>The tool I used has enough features to do the assignment.</td>
</tr>
</tbody>
</table>

10) If you select ii for question 8 please brief the comparison.
Appendix P: Tutorial- Week 9

Web Engineering

Tutorial: Process Modelling

Week 9


3. Compare the visual process-modelling tool you have used in week 8 with state table approach in terms of “easiness to model” according to your experience.