AN EMPIRICAL STUDY: THE METHODOLOGIES, TECHNIQUES, AND TOOLS USED IN THE DEVELOPMENT OF WEB APPLICATIONS AND WEB SITES IN IRELAND.

BY

Sylvia Walsh, B.A.

SUBMITTED IN PARTIAL FULFILMENT

FOR THE

Degree Of Masters Of Business Studies

Of

The Galway-Mayo Institute Of Technology

Head of Department: Mr. Larry Elwood
Research Supervisor: Mr. Kevin Heffernan

Submitted to the Higher Education and Training Awards Council, June 2004

SUMMARY OF TABLE OF CONTENTS

TABLE OF CONTENTS

DEDICATIONS

This thesis is dedicated to my Mom, Dad and Brother for their guidance, support, inspiration, love and enthusiasm.

Also this thesis is dedicated to Padraic whose motivation kept me afloat amidst the storm.

ACKNOWLEDGEMENTS

I gratefully acknowledge the assistance and support of the following people in achieving the completion of this thesis:

Kevin Heffernan, for recommending this research topic and for his tremendous support, guidance and patience throughout this research.

Larry Elwood for affording to me the resources of his department.

David Heffernan, David Martin and Shane McGinley for their assistance.

Annmarie Birkett, Clare O'Connor, Joanne Farragher and Noreen Goggen, for their constant support.

My Mother and Father who taught me that I could accomplish anything that I set my mind to. My Brother who supported me all the way in my studies.

Padraic Flaherty for his constant encouragement and support in various ways during the achievement of this thesis.

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDICES

CHAPTER 1: INTRODUCTION

1.1 Objective Of The Study

This study is generally concerned with the methodologies, tools and techniques used in the development of Web-based applications, with regard to both the current and future trends of the subject developers. It is especially concerned with analysing the effort applied to the use of methodologies for developing Web-based applications.

The study analyses the effort applied to developing Web-based applications, and the perceived effectiveness of methodologies, techniques and tools.

1.1.1 Primary Objectives

The primary objective of this research is as follows:

- To study the methodologies, techniques and tools used in the design and development of Web-based applications.

1.1.2 Secondary Objectives

The secondary objectives of this study are as follows:

- To analyse the efforts applied to developing Web-based applications.

- To determine the perceived effectiveness of methodologies used in the development of Web-based applications.
- To determine major problems incurred by organisations, with regard to the design of their methodologies.

1.2 Introduction To The Literature Review

The purpose of the literature review is to establish the background to the study, which is bounded within the context of information systems development and Web application development. The various dimensions and complexities of Web applications and the methodologies, techniques and tools are explored. The object-oriented paradigm is introduced to facilitate the development of Web-based applications.

The remainder of this chapter presents the research plan by outlining the objectives of subsequent chapters and the strategy employed for achieving those objectives.

1.3 The Research Plan

The layout of the thesis is as follows; Chapter two introduces the concept of information systems. The literature review establishes the contents of information systems. Following is a discussion on types of information systems, and the advantages and disadvantages of information systems.

Chapter three focuses on the process of Information Systems Development (ISD). Traditional approaches to systems development are introduced. Most of these are derived from or based on the System Development Life Cycle (SDLC). This chapter also focuses on the object-oriented development paradigm.

Chapter four presents a discussion on Web application development. The chapter introduces the reader to the basic concept of Web applications. The discussion includes types, architecture and development of Web-based applications.

Chapter five outlines the research methodology used in the course of the study. The research methodology was planned with consideration to time constraints. Case study was chosen as the research instrument due to its suitability for obtaining the type of and depth of information required. Chapter six presents the research findings based on the analysis of the case studies.

1.4 Summary Of Findings And Conclusions

This section provides a summary of the results and the main conclusions drawn from the research.

1.4.1 Summary Of The Findings

NovaVision, Martec and Calico Commerce all state that Web-based applications add a significant contribution to each of the individual companies value chain. All three companies agree that providing Web-based applications as a service increases their potential client base. The range of systems developed by NovaVision and Martec fall between €5,000 - €19,000 in terms of cost to customer bracket. Those developed by Calico Commerce are substantially more expensive. This may be attributed to the fact that Calico Commerce customer base by far exceeds that of NovaVision and Martec.

There is a parallel between Martec and Calico Commerce with regard to their development effort in terms of system design and coding. Both companies allocate 50% of their development time to this area. The research suggests that this may be attributed to the techniques and tools that both companies are using for the development of information systems.

NovaVision, Martec and Calico Commerce have not previously or do not currently use any formal Web specific development methodology in the design and development of Web-based applications or Web sites.

Websphere is the primary development tool used by NovaVision. It accounts for 60% of their development. Dreamweaver is the primary development tool used by Martec for their Web layer development. Dreamweaver accounts for 30% of their development, while Microsoft Access accounts for 20% of their database development. Dreamweaver is the primary development tool used by Calico Commerce. Dreamweaver accounts for 40% of their Web layer development, while Oracle is used for 35% of their database development.

The software process combined with the techniques that NovaVision use suggests that they may be applying the Hypermedia Flexible Process Modelling (HFPM) approach (Olsina, 1998). The approach used by Martec, combined with the techniques that the use, suggest that they may be following the Lowe-Hall's Engineering Approach (Lowe et al., (1999).

1.4.2 Summary Of Conclusions

The research indicates that the consensus for Web specific methodologies has not yet reached general acceptance. With regard to NovaVision, Martec and Calico Commerce the following conclusions were drawn:

- The vastly published and universally available methodologies are not used in the entirety, if they are used at all.
- They develop in-house methodologies based on best practices, which have evolved.

The research suggests expectations for high levels of usage of Web specific methodologies is further reduced by the following:

- Only one of the companies interviewed was capable of identifying some of the published Web-based application development methodologies.
- One of the respondents admitted that perhaps the methods they are using, fall into a category of a Web specific development methodology.

The study of NovaVison, Martec and Calico Commerce identified that deficiencies exist in the development of Web applications, especially in the areas of t*echnology*, *guidelines,* and *documentation.* All companies feel that the issue of Web specific methodologies is over-hyped and expressed doubt over the academic papers put forward by Web Engineers.

CHAPTER 2: INFORMATION SYSTEMS

2.1 Introduction To Information Systems

O'Brien (1999) provides the following definition of a system:

> *"A system is a group of interrelated components working together toward a common goal by accepting inputs and producing outputs in an organised transformation process."*

A system has three basic components. Input involves capturing and assembling elements that need to be processed. Processing converts inputs into outputs. Outputs involve transferring elements that have been produced by a transformation process to their ultimate destination.

O'Brien (1999) defines an information system as:

> *"An information system is an arrangement of people, data, processes, interfaces, networks, and technology that interact to support and improve both day-to-day operations in a business."*

The components of information systems are people, hardware, software, data and network resources to perform input, processing, output and storage. Information systems control activities that transform data resources into information products.

People are required for the operation of information systems. Hardware includes the physical devices used to transform data. Software is used to direct all sets of information processing instructions.

Information systems are designed to provide crucial information to users for decision-making. Information needs to be available at the correct time, and at the appropriate level of detail to be of use to its recipient (Avison et al., 1998).

Buckingham et al., (1978) as cited in Avison et al., (1995) defines an information system as:

> *"A system which assembles, stores, processes and delivers information relevant to an organisation, in such a way that the information is accessible and useful to those who wish to use it, including managers, staff, clients and citizens. An information system is a human activity (social) system which may or may not involve the use of computer systems."*

Lucey (1991) defines an information system referencing the fact that it is part of a wider management system.

> *" It must provide support and assistance to management for planning, control, decision making and other functions."*

2.2 Goals Of Information Systems

Information systems may assist an organisation to achieve improved efficiency of its operations and improved effectiveness through enhanced managerial decisions. Information systems are regarded as providing a competitive advantage for an organisation (Avison et al., 1998).

Information systems can increase operational efficiency by completing many of the standard tasks, in a manner that is superior, more economical, and faster than the systems that precede it. A Decision Support System provides a set of easy to use modelling, retrieving, and reporting capabilities so that users can generate the information they feel will be useful to them when making decisions.

Whether executives or senior managers are computer literate or not, research indicates that Executive Information Systems assist managers to improve the speed and effectiveness of decisions. According to O'Brien (1999) information systems perform three vital roles in any type of organisation; firstly they support business operations; secondly they support managerial decision-making; and thirdly, they support strategic competitive advantage.

2.3 Contents Of An Information System

Parker et al., (1993) identify five elements of an information system: *hardware, software, procedures, people, and data.* This section is concerned with people as information system users (also known as end users) and with procedures that govern the operation of a computer system. A common analogy that is used to illustrate the role of procedures in an information system is as follows *"Procedures are to people what software is to hardware"* (Parker et al., 1993).

2.3.1 Hardware

Gallagher et al., (1997) define hardware as follows:

> *"Hardware refers to all the physical components that make up a computer system."*

The hardware of a computer is the physical components of the system. It consists of input devices, processors, storage devices and output devices. Morgan et al., (1996) identify the following input hardware: *keyboard, mouse, scanner, light-pen, magnetic card reader, voice data entry, optical character recognition, magnetic-ink character recognition,* and *touch sensitive screen.* Gallagher et al., (1997) identify the following output hardware: printers such as *impact printers, non-impact printers, laser printers* and the *visual display unit.* Devices such as modems and touch screens, provides both input and output capabilities.

2.3.2 Software

Morgan et al., (1996) provides the following definition of software:

> *"Software is a set of instructions that enables the computer to perform its function."*

Bocij et al., (1999) define software as follows:

> *"Software can be defined as series of detailed instructions that control the operation of a computer system. Software exists as programs that are developed by computer programmers."*

Software can be categorised into two types: *system software* and *application software.* System software manages and controls the operation of the computer. Operating systems are software that interacts with the hardware of the computer in order to manage and regulate the computer resources.

2.3.3 Procedures

Procedures are the policies that control the operation of a computer system. Procedures describe how end-users interact with the system. Information systems generally process data into information using one of three methods, *batch, on-line, or real time* processing (Lucey, 1991).

Batch processing involves individual source documents being grouped or batched manually or on a computer and processed at a predetermined interval of time. *On-line processing* involves processing data as it occurs. *Real-time processing* involves immediate or rapid response after a change takes place.

2.3.4 End Users

O'Brien (1999) defines an end-user as:

> *"Anyone who uses an information system or the information it produces."*

End-users are a vital component in the development of information systems as they identify and establish the requirements of a new system. The end-user is the focal point of an information system. End-users specify the type of hardware and software to meet their needs, develop the system, and define the appropriate security and backup procedures of the system.

Satzinger et al., (2002) state that end user activities include the following; *creating records or transactions*, *modifying database contents*, *generating reports, querying databases, importing or exporting data.* Hackathorn (1988) as cited in Bocij et al., (1999) defines end-user computing as the following:

> *"An information processing activity in which the person has direct personal control over all stages of the activity."*

2.3.5 Data And Information

The terms data and information are often used synonymously. It is important to note that there is a distinctive and significant difference. Data consists of raw facts or observations that are considered to have modest or no value until they have been processed and transformed into information (Bocij et al., 1999).

Information is data that has been processed in such a way that it is useful to the recipient. *"Data are processed in some way to form information but the mere act of processing data does not itself produce information"* (Lucey, 1991). Information is data processed for a purpose (Curtis et al., 2002).

2.3.5.1 Two Paradigms Of Information

There are two contrasting paradigms which describe the nature and value of information (Flynn, 1998) and are known as *The Resource-Driven Paradigm* and the *Perception-Driven Paradigm.*

The Resource Driven Paradigm views information as a crucial organisational resource which needs to be managed. The information exists independently of the receiver and it does not change during transmission (Flynn, 1998).

The Perception-Driven Paradigm, in contrast assumes that information does not exist beyond the perceptions of the receiver. Interpretations will vary according to different psychological and social factors, information is transient and subjectively determined (Flynn, 1998).

2.3.5.2 Characteristics Of Information

Information can be said to have a number of different characteristics that can be used to describe quality. The difference between good and bad information is identified by considering the fact of whether or not it has some or all of the attributes of information quality. O'Brien (1999), takes an organised approach and describes the attributes of information quality as being divided into three basic dimensions; *time, content* and *form*. See Table 2.1.

Table 2.1: Characteristics Of Information

Time Dimension	
Timeliness	Information should be provided when it is needed.
Currency	Information should be up-to-date when it is provided.
Frequency	Information should be provided as often as needed.
Time Period	Information can be provided about past, present, and future time periods.
Content Dimension	
Accuracy	Information should be free from errors.
Relevance	Information should be related to the information needs of a specific recipient for a specific situation.
Completeness	All the information that is needed should be provided.
Scope	Information can have a broad or narrow scope, or an internal or external focus.
Performance	Information can reveal performance by measuring activities accomplished, progress made, or resources accumulated.
Form Dimension	
Clarity	Information should be provided in a form that is easy to understand.
Detail	Information can be provided in detail or summary form.
Order	Information can be arranged in a predetermined sequence.
Presentation	Information can be presented in narrative, numeric, graphic or other forms.
Media	Information can be provided in the form of printed paper documents, video displays, or other media.

Table 2.1: Adapted from O'Brien 1999

Information should assist in the reduction of uncertainty, and should help to make more informed decision-making. Avison et al., (1998) discuss the following five characteristics of information.

- Information must be timely with respect to its intended use.
- Information must be appropriate for the type of task being undertaken and the personnel involved.
- Information must be in a form that is understandable to the target recipients.
- The degree of accuracy of the information must be appropriate for its usage.
- Information calling for action must be directed to the person who can initiate the appropriate action.

2.3.5.3 Information And Decision Making

Knowledge of managerial activity is a precondition for effective systems design and implementation (Gorry et al., 1971). Decisions that are made at an operational management level tend to be structured and short term. Decisions made at the tactical level tend to be semi structured and span from short-term to medium term plans. Decision-making at strategic level tends to be more unstructured. These decisions tend to be long term and prognostic. See Table 2.2.

2.3.5.4 Information Requirements By Decision Category

Gorry et al., (1971) expound the ideas of Anthony (1965) to present a constructive technique of categorising decisions by comparing managerial activities against the degree of structure in a decision. See Table 2.3.

Table 2.2: Information And Decision Making		
Management Level	**Decision Characteristics**	**Information Characteristics**
Strategic	Long time horizons, large scale resources, much creativity and judgement, usually unstructured, problems difficult to define, infrequent, much uncertainty.	Largely external, informal resources important, forward looking, qualitative information important, precision unimportant. Instant access not vital, wide-ranging, incomplete.
Tactical		
Operational	Repetitive, short time scale, small scale resources, usually structured, clear objectives and decision rules, little or no discretion.	Largely internal, mainly historical, detailed, often quantitative, high precision, instant availability often critical, narrow in scope, comprehensive.

Table 2.2: Adapted from Lucey, (1991)

Table 2.3: Information Requirements By Decision Category

Characteristics of Information	Operational Control	Management Control	Strategic Planning
Source	Largely Internal	⟷	External
Scope	Well defined, narrow	⟷	Very Wide
Level of Aggregation	Detailed	⟷	Aggregate
Time Horizon	Historical	⟷	Future
Currency	Highly Correct	⟷	Quite Old
Required Accuracy	High	⟷	Low
Frequency Of Use	Very Frequent	⟷	Infrequent

Table 2.3: Adapted from Lucey (1991)

2.4 Types Of Information Systems

Anthony's (1965) three layer model of the organisation provides the basis for the most commonly used categorisation in information systems. It consists of operational control, tactical planning, and strategic planning. However, due to the fact that operational control involves monitoring the routine operations of the firm, this model translates into a four layer categorisation of information systems within the organisation as depicted in Figure 2.1.

Figure 2.1: Types Of Information Systems

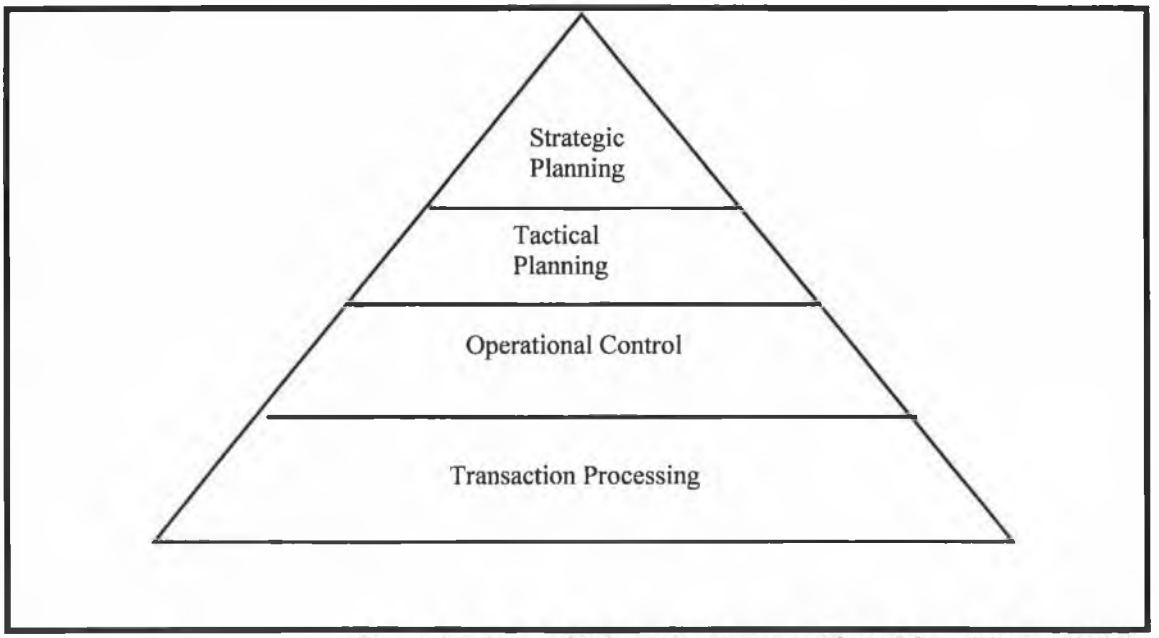


Figure 2.1:Types of Information Systems, Adapted from O'Brien (1991)

2.4.1 Transaction Processing Systems

Transaction Processing Systems (TPS) automate the handling of data concerning business activities or transaction, which can be thought of as simple, discrete events in the life of an organisation (Hoffer et al., 2002).

Transaction Processing Systems (TPS) computerise the everyday accounting and administrative functions of the organisation. Transaction processing is vital to keep the operations of the organisation running efficiently and provides the foundation for all other internal information support. These systems are pre-specified and are directly related to the structure of the organisations data.

2.4.2 Operational Control Systems

These are also known as Management Reporting Systems (MRS). These are used both in management planning and management control functions. Operational control systems offer information to management in a predefined format. These systems occur at lower levels of the organisation and are based on unambiguous, clear data of a quantitative or financial nature.

Reports produced by Management Reporting Systems (MRS) can fall into one of three categories: *Schedule Reports*, sometimes called periodic reports, *Exception Reports*, are issued when something unexpected occurs, and *Demand Reports*, provide management with reports when requested. Most reports generated by an MRS are either scheduled reports or exception reports (Parker et al., 1993).

2.4.3 Planning And Analysis Applications/Decision Support Systems

Keen et al., (1978) provide the following definition of a Decision Support System (DSS).

> *"A DSS is a coherent system of computer-based technology (hardware, software and supporting documentation) used by managers as an aid to their decision making in semi structured decision tasks."*

Keen et al., (1978) describe Decision Support Systems as the use of computers to:

- Assist managers in the decision making process for semi structured tasks.

- Support rather than replace managerial judgement.
- Improve the effectiveness of decision making rather than its efficiency.

Decision Support Systems (DSS) are designed to assist the effectiveness and productivity of managers and professionals. (Keen, 1981) conducted an interesting study of a number of organisations regarding their Decision Support System use. He concluded that the decision to build a Decision Support System appeared to be based on value rather than cost. He outlined the benefits of a Decision Support System as the following:

- Amplify the number of alternatives examined.
- Superior understanding of the business.
- Faster response to unexpected situations.
- Ability to carry out ad hoc analysis.
- New insights and learning.
- Improved communication, control, cost saving and better decisions.
- More effective teamwork.
- Time saving and making better use of data resources.

The expansion of Decision Support Systems, accompanied with future trends have resulted in a variety of Decision Support Systems. Examples include Group Decision Support Systems (GDSS) and Executive Information Systems (EIS).

Group Decision Support Systems support the activities and decision making of an entire team, rather than just individuals. Executive Information Systems (EIS) assist top management by providing information on essential areas of the organisation activities from both internal and external databases.

Executive Information Systems have proven that they provide senior managers access to the information they require, which is extracted from the wealth of available computer-based data. An Executive Information System can aid managers by delivering and displaying organisational and external data in ways that each individual understands.

2.4.4 Strategic Applications

O'Brien (1999) defines a Strategic Information System as follows:

> *"Information systems that support or shape the competitive position and strategies of an enterprise."*

Strategic Information Systems can be used to secure competitive advantage in the marketplace.

Strategic Information Systems can offer a competitive advantage: *create barriers to competitors, building of customers switching costs and or operational dependence, change the basis of competition by offering new services, products, or information not offered by competitors, changing the organisations operations in such a way as to change the nature or environment of the business and precipitating a business process* (O'Brien, 1999).

Strategic Information Systems (SIS) are used by executives of an enterprise. They are concerned with decisions that affect the long-term policy of the organization (Cleary, 1998).

2.4.5 Distributed Information Systems

Tanebaum et al., (2002) provide the following definition of a distributed system:

> *"A distributed system is a collection of independent computers that appears to its users as a single coherent system."*

A Distributed Information System is a collection of autonomous computers connected by a computer network to facilitate resource sharing and co-operation between applications to complete a given assignment.

Distributed Information Systems should hide where conceivable, the distributions of information, giving the impression to users that all the accessible resources are located at the users workstation.

Date (1995) discusses the fundamental principle of a distributed database as follows:

> *"To the user, a distributed system should look exactly like a non distributed system."*

Users of a distributed system should be able to behave exactly as if the system were not distributed. Users and applications can interact with a distributed system in a consistent and uniform way, regardless of where and when the interaction takes place. Distributed systems should be relatively effortless to expand or scale. A distributed system will normally be continuously available, however certain parts may be out of order. Users and applications should not notice parts are being fixed or replaced, or that new parts are being added to serve more users and applications (Simon, 1996).Tanebaum et al., (2002) state that Distributed Information Systems should, easily connect users to resources, hide the fact that resources are distributed across a network, i.e. be transparent, be open, and be scalable.

2.4.5.1 Centralised Verses Distributed Systems

A historical perspective on the evolution of distributed systems discloses a multitude of advantages and disadvantages. A centralised system can be equally reactive to end-user needs in the correct situation as well as offering superior security, data integrity and systems management functionality.

2.5 Advantages Of Distributed System

The principal advantages of distributed systems is their efficiency to permit the sharing of information and resources over an extensive geographic area, enabling a systems designer to have freedom to optimise the placement of distributed system components (Simon, 1996). Therefore the following advantages are supported:

2.5.1 Improved Flexibility

Computers and other IT infrastructure components can be flexibly located at points within the organisation where they can be utilised most effectively. Components can be added, upgraded, moved and removed without impacting upon other components, to meet present and future needs.

2.5.2 Local Autonomy

Date (1995) defines local autonomy as follows:

> *"The capability to administer a local database and to operate independently when connections to other nodes have failed."*

With local autonomy, each site has the capability to control local data, administer security, log transactions, and recover when local failures occur, and provide full access of local data to local users when any central or coordinating site cannot operate.

2.5.3 Increased Reliability And Availability

In a centralised system, a component failure can mean that an entire system fails, and the IT function is unavailable to the organisation. In a distributed system, multiple components of the same type can be configured to fail independently and provide a level of fault tolerance. Therefore, a distributed system will continue to function at some reduced level, even when a component fails.

2.5.4 Faster Response

Depending on how data are distributed, most requests for data by users at a particular site can be satisfied by data stored at that site. This speeds up query processing since communication and central computer delays are minimised.

It may also be possible to split complex queries into sub queries that can be processed in parallel at several sites, providing even faster response.

2.5.5 Security Breaches Are Localised

A security breach in one domain of a distributed system does not compromise the entire system. Each security domain has varying degrees of security authentication, access control and auditing.

2.6 Disadvantages of Distributed Systems

Distributed systems also have some disadvantages, most of which correspond to the advantages of centralised systems. Simon (1996) discusses the following:

2.6.1 Difficult To Manage And Secure

Centralised systems are inherently easier to secure and easier to manage. Distributed systems require more complex procedures for security, administration, maintenance and user support due to greater levels of co-ordination and control required.
This usually results in higher costs associated with managing and securing a distributed systems environment.

2.6.2 Reduced Reliability And Availability

In contrast to the potential improvements in reliability and availability discussed above, a distributed system consists of many components, which can potentially fail, causing loss of availability.

2.6.3 A Shortage Of Skilled Support And Development Staff

In a decentralised operation, scarce support and development can be dispersed, resulting in a loss of economies of scale, which in turn, leads to higher costs. Another issue is the level of support offered by vendors.

The commitment and level of support offered by vendors of distributed software and hardware to organisations working with the issues of building large-scale distributed systems is not yet comparable to the traditional large mainframe-based vendors (Simon, 1996).

The development of distributed systems entail many complexities, which will be discussed later in Chapter 4.

2.7 Summary

Information systems are designed to support business operation and managerial decision-making. Hardware, software, end-users, procedures and data are all critical components of information systems. The order in which a developer perceives information and its interpretation will affect the developers approach when developing information systems. Contrary to the fact that there are numerous benefits associated with information systems, there are many problems associated with distributed information systems.

The problems include: difficulty to manage and secure, reduced reliability and availability and a shortage of skilled support and development staff.

2.8 Conclusions

A wide variety of organisations are deploying information systems on the World Wide Web. The World Wide Web is being used as a strategic business tool, supporting existing operations, and providing a low cost solution for delivering services on line.

The principles behind the methodologies and techniques used in the development of traditional information systems can be applied to the development of Web-based information systems.

However, relevant adjustments need to be applied to the development methodologies, as the World Wide Web requires certain technological specifications.

This chapter has reviewed the categories of Information Systems and both the advantages and disadvantages of Information Systems. The next chapter will review the methodologies, techniques, and tools used in the development of information systems.

CHAPTER 3: INFORMATION SYSTEMS DEVELOPMENT

3.1 Introduction

Russo et al., (2002) define information systems development as follows:

> *"Information systems development is a fundamental process performed when engaging information technology to achieve a specific purpose in a specific context, although it involves much more than simply the deployment of technology. It is an activity that is fundamental in that it is stable over time even though the context in which it is performed changes."*

Whitten et al., (2001) defines the system development process as follows:

> *"A set of activities, methods, best practices, deliverables, and automated tools that stakeholders use to develop and maintain information systems and software."*

Hirschheim et al., (1995) state that information systems development (ISD) is a socio-technical discipline. Social and technical elements are significant. Information systems have a very strong social element. While social factors are critically important, it is the technology that serves to uniquely characterise the information systems field. Information systems development (ISD) is largely based on the reuse of development patterns that are recognised by developers.

Hirschheim et al., (1995) defines information system development methodologies as follows:

> *"An information systems development methodology is an organised collection of concepts, methods, beliefs, values and normative principles supported by material resources."*

Information systems development methodologies strive to shape the process of developing information systems in a straightforward and simple manner. They produce a set of procedures, techniques, and tools, which must be followed. This in turn, enables the process of managing and developing information systems more efficiently and effectively.

There is no single methodology that covers all aspects of systems development, however some methodologies are more comprehensive than others. Hirschheim et al., (1995) state that the evolution of information systems development methodologies have taken the form of seven overlapping stages or generations;

- *First Generation*, the emergence of formal life-cycle approaches.
- *Second Generation*, the emergence of the structured approaches.
- *Third Generation*, the emergence of prototyping and evolutionary approaches.
- *Fourth Generation*, the emergence of socio-technical, participative approaches.

- *Fifth Generation,* the emergence of sense-making and problem formulation approaches.
- *Sixth Generation,* the emergence of the trade-union led approaches.
- *Seventh Generation,* the emergence of emancipatory approaches.

The following is a brief overview of ETHICS (Effective Technical and Human Implementation of Computer Systems) and SSM (Soft Systems Methodology).

3.1.1 ETHICS (Effective Technical and Human Implementation of Computer Systems)

User participation is fundamental at every stage of ETHICS. The entire process can be considered as a method of balancing the costs and benefits of social and technical solutions upon job satisfaction and efficiency (Beaumont, 2003).

The ETHICS methodology contains six stages which are further divided into twenty-five steps (Mumford, 1983).

Stage 1: Essential Systems Analysis (Steps 1 to 11).

Stage 2: Socio-Technical Systems Design (Steps 12 to 20).

Stage 3: Setting out Alternative Solutions (Steps 21 to 23).

Stage 4: Setting Out Compatible Solutions (Step 23).

Stage 5: Ranking Socio-Technical Solutions (Step 24).

Stage 6: Prepare A Detailed Work Design (Step 25).

The common needs are agreed by forming a 'core' module of the proposed information system, delivering essential information needs in the group-decision process (Avison et al., 1995).

3.1.2 SSM (Soft Systems Methodology)

Soft Systems Methodology was developed by Peter Checkland. He saw how there were problems associated with complex systems which had a large social components. Checklands "Soft Systems Methodology" was developed through a series of research projects. The methodology was published in 1981.

Soft Systems Methodology is divided into 7 distinct stages;

Stage 1: Problem situation: Unstructured.

Stage 2: Problem situation: Expressed.

Stage 3: Naming of Relevant Systems.

Stage 4: Comparing Conceptual Models with Reality.

Stage 6 & 7: Implementing 'Feasible and Desirable' changes.

Socio-technical systems have had little influence to date in the development of Web based applications, therefore for the purpose of this study techno-centric methodologies will be discussed.

3.2 Recommendations For Successful Development

Benjamin (1971) as cited in Whitten et al., (2001) put forward recommendations for successful management of the systems development process. The following recommendations which are endorsed by Whitten et al., (2001) are presented here: *Get the owners and users involved, Use a problem solving approach, Establish phases and activities, Establish standards, Justify systems as capital investments, Cancel or revise scope, Divide and conquer, Design systems for growth and change.*

3.2.1 Get The Owners And Users Involved

Hoffer et al., (2002) defines end-users as follows:

> *"The people for whom the systems are designed and located in the functional departments."*

The end user has a vital role to play in the success or failure on a new information system. Information systems need to meet social and technical needs. In order to achieve this, it is necessary for end-user participation. Owner and user involvement as well as education all assist with the acceptance of a new system.

Individuals responsible for systems development must insist on participation of owners and users (Whitten et al., 2001).

3.2.2 Use A Problem Solving Approach

A methodology is a problem solving approach to building systems. Trepper (2000) defines a methodology as the following:

> *"A methodology in general is a set of steps designed to guide the development process from the beginning of a project to production turnover and maintenance."*

Whitten et al., (2001) put forward the classic problem-solving approach.

1. Study and understand the problem and its content.
2. Define the requirements of a suitable solution.
3. Identify candidate solutions and select the best solution.
4. Design and/or implement the solution.
5. Observe and evaluate the solutions impact, and refine the solution.

3.2.3 Establish Phases And Activities

The software development process includes analysis, design, implementation and maintenance. However, due to the size of many software development projects, it is imperative to divide the phases into discrete tasks. From there, the relevant methods and techniques can be applied in relation to each project.

3.2.4 Establish Standards

It is necessary for an organisation developing information systems to embrace the standards and the process used to develop systems. Whitten et al., (2001) state that standards should include the following; *Documentation, Quality, Automated Tools* and *Information Technology*.

- Documentation should be a by-product of the systems development effort. It should help to open all phases rather than being viewed as a single phase.
- Quality ensures that the deliverables of a phase or activity meet expectations.
- Automated tools assist in the development and maintaining of information systems.
- Information technology directs information systems to a common configuration.

3.2.5 Justify Systems And Capital Investments

Information systems are capital investments. When considering a capital investment, it is important to address two problems. Firstly the system analyst must adopt the best solution, not just the first solution that is presented. Secondly the system analyst should evaluate a solution in terms of cost-effectiveness and risk management.

3.2.6 Cancel Or Revise Scope

The phased approach to systems development provides opportunities to re-evaluate feasibility. This is an advantage. Whitten et al., (2001) note that in the long term, it is more viable to cancel a project, which will not meet the required needs of the users, rather than to implement a project,

3.2.7 Divide And Conquer

It is imperative for system analysts to note that in order to conquer the problems of building large systems, it is necessary to divide them into subsystems. This enables the analyst to simplify the problem-solving approach.

3.2.8 Design Systems For Growth And Change

Organisations are constantly changing and therefore their needs in terms of information systems will also be changing. It is important for organisations not to fall into the trap of developing systems to meet the system requirements of today only. Systems must be flexible enough to meet anticipated future requirements.

3.3 The System Development Life Cycle

The information systems development life cycle is considered to be a model of the stages in the life of an information system. The life cycle is the most basic method that has been applied to the majority of information systems development programs (Avison et al., 1998).

The systems development life cycle enables all the organisational resources to be applied to the development process in an effective and timely manner (Bronzite 2000). The system development life cycle was developed to produce information systems on schedule, within a required budget, and to the requirements of the user.

In large systems, management decides the merit of a particular project. They help to formalise the concept of user involvement in the development process, as well as ensuring that all activities are satisfying the information requirements.

Russo et al., (2002) identify a number of characteristics inherent in the system development life cycle: The life-cycle consists of discrete stages, each of which has distinct activities. There is a *stage-limited commitment*, in that the end of one stage the decision to proceed is only valid until the completion of the next stage. At the end of each stage, there is a *signoff of interim end-products.* Whitten et al., (1998) offer the following definition of the system development life cycle (SDLC):

> *"A logical process by which systems analysts, software engineers, programmers, and end users build information systems and computer applications to solve business problems and needs."*

The most basic life-cycle is composed of three stages, systems analysis, system design and system construction. There are many variations, all of which are derived by decomposing the processes of analysis, design and construction into sub-processes. O'Brien (1999) describes a five phase model, systems investigation, systems analysis, systems design, systems implementation, and system maintenance.

For the purpose of this study, the life-cycle has been compressed into three phases, systems analysis, encompassing the initial investigation and detailed analysis, systems design, and systems construction encompassing both implementation and maintenance.

Table 3.1 depicts from left to right the three major stages of the information systems development life-cycle as referenced in this study. The latter are subsequently subdivided into eleven discreet tasks as identified by O'Brien (1991).

Table 3.1: An Illustration Of The Traditional Systems Development Life Cycle	
Three Stages Model of the SDLC	**Eleven Tasks of the SDLC**
Systems Analysis **Product:** Feasibility Study And Systems Requirements	• Determine whether a business problem or opportunity exists. • Conduct a feasibility study to determine whether a new or improved information system is needed. • Develop a project management plan and obtain approval. • Analyse in detail the information needs of end users, the organisational environment, and any system presently used.
System Design **Product:** System Specification	• Develop the logical input, processing, output, storage and control requirements of a system that can meet the needs of end-users. • Develop specifications for the hardware, software, people, data resources, and information products that will satisfy the information needs of end-users.
System Construction **Product:** Operational And Evolving Systems	• Acquire (or develop) and install hardware and software. • Test and document the system. • Train people to operate and use the system. • Convert to the new system. • Use a post implementation review process to monitor, evaluate, and modify the system as needed.

Table 3.1 Adapted from O'Brien (1991)

3.3.1 Phases Of The System Development Life Cycle

The concept of the systems development life cycle (SDLC), according to O'Brien (1999) considers the information system development process as '*highly related and interdependent.*' The development activities can occur at identical times, therefore, different sections of a development project can be at different stages of the development cycle.

3.3.1.1 System Analysis

System analysis is carried out in detail to determine the requirements of the new system. This phase results in a detailed description of the system, which is required by the user. This phase requires that each problem is understood, and that the most fitting solution is selected. Parker (1998) defines systems analysis as follows:

> *"Systems analysis is the phase of systems development in which a problem area is studied in depth and the needs of system users are assessed."*

Hirschheim et al., (1995) discusses systems analysis in term of process:

> *"Systems analysis is the process of collecting, organising, and analysing facts about a particular information system and the environment in which it operates."*

The purpose of this process is to develop a comprehensive understanding of the business, the existing information systems, and to evaluate and identify the problems and opportunities. It is noted if there is a shortfall between the existing system and the required system. From there the new system to be implemented is planned and presented.

Systems analysis begins with a preliminary feasibility study, which is conducted to decide whether or not a new system is feasible. The feasibility study examines the technical feasibility, operation feasibility, economic feasibility, and schedule feasibility, all of which determine whether the costs and benefits justify a new system. If a current system exists, this is examined in detail for the purpose of producing a document of requirements for the new system. In a situation where the organisation does not have an existing information system, the development of the requirements document commences from a more conceptual basis. In either case, this document should incorporate the concepts and outputs depicted in Table 3.2.

Table 3.2: Outputs From System Analysis	
Phases Of System Analysis	**Outputs**
Survey Phase	Problem statement. Scope statement. Project plan. Project charter.
Study Phase	System Models. Process analysis models/Process analysis data. Cause effect analysis. System improvement objectives and constraints. Revised project plan. Detailed study findings.
Definition Phase	Requirement statement outline. System models. Discovery prototypes. Business requirements proprieties. Revised project plan.

Table 3.2 adapted from Whitten et al., (2001)

3.3.1.2 System Design

Systems analysis focuses on what the new system should do. Systems design focuses on how the new system should be built. Whitten et al., (1998) describe systems design as follows:

"The evaluation of alternative solutions and the specification of a detailed computer-based solution."

System design consists of developing a model of the new system and performing a detailed analysis of the costs and benefits of the new system (Parker, 1998). The process of system design commences after there is an understanding of the knowledge gained from the systems analysis phase. The system analysis phase is the foundation of the design of the new system. The document of requirement is therefore the basis for designing the new system.

During the design phase the system designer must work with the end users to determine the system input, output, and processing methods, develop the specifications of the system for programmers, and present the system design to the management and end users for their approval. The model of the new system should encompass the following. See Table 3.3.

Table 3.3: Outputs From System Design	
Phase Of System Design	**Outputs**
The Configuration Phase	Candidate solutions. Feasibility analysis. System proposal.
The Procurement Phase	Potential vendors, product options, and technical criteria. Request for proposal. Validated proposals. Hardware and software recommendations. Integration requirements.
Design And Integration	Normalised distributed data models. Distributed process models. Database design specification. Input and Output design specifications. Interface design specification. Technical design statement.

Table 3.3 adapted from Whitten et al., (2001)

3.3.1.3 System Construction

This phase constitutes the construction of the new system and the delivery of the system into day-to-day operations (Avison et al., 1998). Software construction should be preceded by the preparation of the requirement specification document (Norman, 1996). This is a highly technical and time consuming process, which encompasses the following tasks:

Implementation Tasks

- Writing, testing, debugging and documenting programs.
- Converting data from the old to the new system.
- Training the system users.
- Installing the new hardware required by the system.
- Developing operating procedures for the computer centre staff.
- Completing systems documentation.
- Evaluating the final system against user requirements.

Maintenance Tasks

- Resolving necessary changes and correcting errors.
- Enhancing or modifying the system.

3.3.2 Strengths Of The System Development Life Cycle

The system development life cycle executes in a sequential fashion. Flynn (1998) suggests that the model oversimplifies actual practice, however due to the fact that it is a linear model, it can offer advantages, such as being clear-cut as each phase has a clear beginning and end.

(Walters et al., 1994) suggest that the use of documentation standards help to ensure that system proposals are complete and that communication is facilitated between users and analysts.

The inclusion of a feasibility study, attempts to assess the costs and benefits of alternative proposals, which enables management to make informed choices.

Flynn (1998) identifies another advantage from the linear nature of the system development life cycle (SDLC). It is due to the fact that analysis and design are completed before construction commences. Otherwise some parts of the system might not fit with others, requiring redesign and recoding.

3.3.3 Weaknesses Of The System Development Life Cycle

There are numerous problems associated with the system development life cycle. Lee (1987) states that the system development life cycle is costly, complex, and a time-consuming process.

Laudon et al., (1995) identify major weaknesses in the system development life cycle. They include the following;

- The conventional development cycle is costly.
- The time needed to develop a system through the life cycle is often lengthy and prolonged.
- The life cycle methodology is relatively inflexible.
- The life cycle methodology discourages change.
- The life cycle methodology is ill suited to the development of decision-oriented applications.

The system development life cycle does not provide a gap between analysis, design, and implementation, which in turn leads to the development of systems that do not meet the requirements of the user (Walters et al., 1994). The System Development Life Cycle (SDLC) is clumsy and outdated. It only pays service to the end user.

3.4 Information Systems Perspective And Their Related Tools

Event partitioning recognises that "*all systems have some functional and some data and some time-dependent behaviour*" (Yourdon, 1986). Yourdon envisages a three dimensional space where the X-axis represents increasing complex functions within a system, the Y-axis represents increasing complex information and the Z-axis represents increasingly complex time-dependent behaviour. Allen (1991) visualised an event partitioning toolset as the following;

- X-axis: Data Flow Diagrams (DFD's)
- Y-axis: Entity Relationship Diagrams (ERD's)
- Z-axis: Event List
- X-Y plane: Entity-Process Matrices
- X-Z plane: State Transition Diagrams (STDs)
- Y-Z plane: Entity State Transition Diagrams (ESTD's)

The three basic perspectives including their related tools are described below; the logical process perspective and Data Flow Diagrams, the logical data perspective and Entity Relationship Diagrams, the logical event perspective and Event Lists.

3.4.1 The Logical Process Perspective

The process perspective was the earliest perspective to be recognised during the evolution of information system methodologies (Olle et al., 1991).

The process perspective is represented by the use of Data Flow Diagrams. McLeod (1993) expresses the following:

> *" A Data Flow Diagram is the most natural way to document processes."*

Data Flow Diagrams were introduced in two seminal books (DeMarco, 1979, and Gane and Sarson, 1979). They became central to most structured methodologies. (Skidmore 1994). Data Flow Diagrams are excellent for documenting existing systems during system analysis as well as documenting new systems during systems design. Hoffer et al., (2002) defines Data Flow Diagrams as follows:

> *"A Data Flow Diagram is a graphical tool that allows analysts to depict the flow of data in an information system."*

The Data Flow Diagram enables structure, which is one of the most important requirements of structured systems. Gane (1990) states that the purpose of a Data Flow Diagram is to show where the data is stored, what processes transform the data, and the interactions between the data stores and processes. Britton et al., (1997) state that Data Flow Diagrams identify the system boundary and external entities and the data or information flows into and out of the system.

3.4.1.1 Functional Decomposition

Functional decomposition as documented by Hoffer et al., (2002) is an *"iterative process of breaking the description of a system down into smaller details."* Functional decomposition is the dividing of a system into components based on subsystems that in turn are further divided. Functional decomposition is illustrated in Figure 3.1. Each numbered box corresponds to a named process. Each group of sub processes within the same level stem from a parent process.

Figure 3.1 depicts decomposition of a process called *order process* into several subprocesses; a credit check process, a select goods process and a schedule order process. The select goods process is further exploded into another layer or level containing two subprocesses; a check inventory process and a requisition goods process. The numbering scheme adapted in the decomposition diagram is used to reference subsequently produced Data Flow Diagram.

Figure 3.1: Functional Decomposition

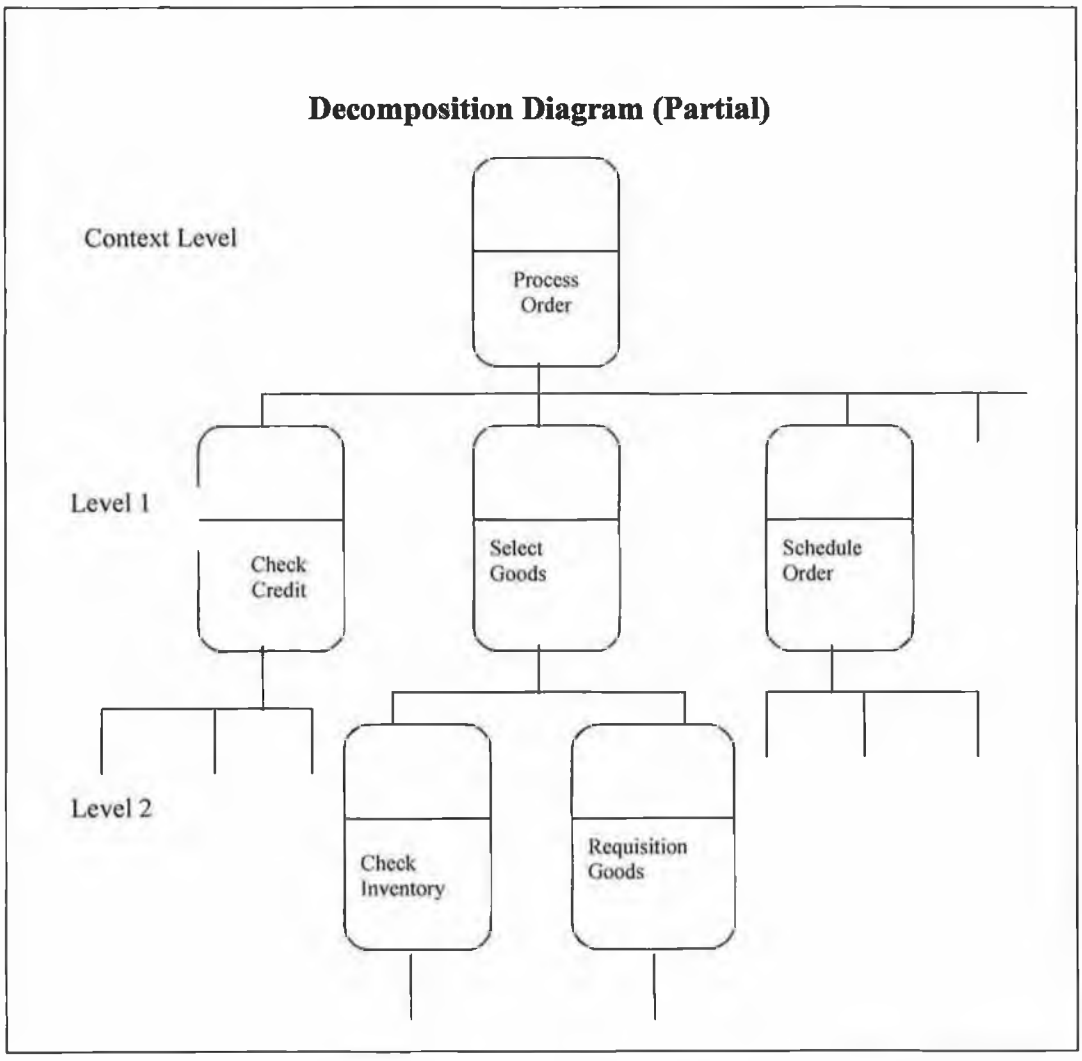


Figure 3.1 Adapted from Whitten et al (2001)

3.4.1.2 Data Flow Diagrams: Symbols And Rules

There are four basic symbols that Data Flow Diagrams use to model the system. They are *Data Flows, Data Store, Processes*, and *External Entities*. There are two basic sets of symbols that are used in the design of Data Flow Diagrams. The first is attributed to Gane and Sarson (1979) and the second attributed to DeMarco and Yourdon (1978). See figure 3.2.

Figure 3.2: DFD Symbols

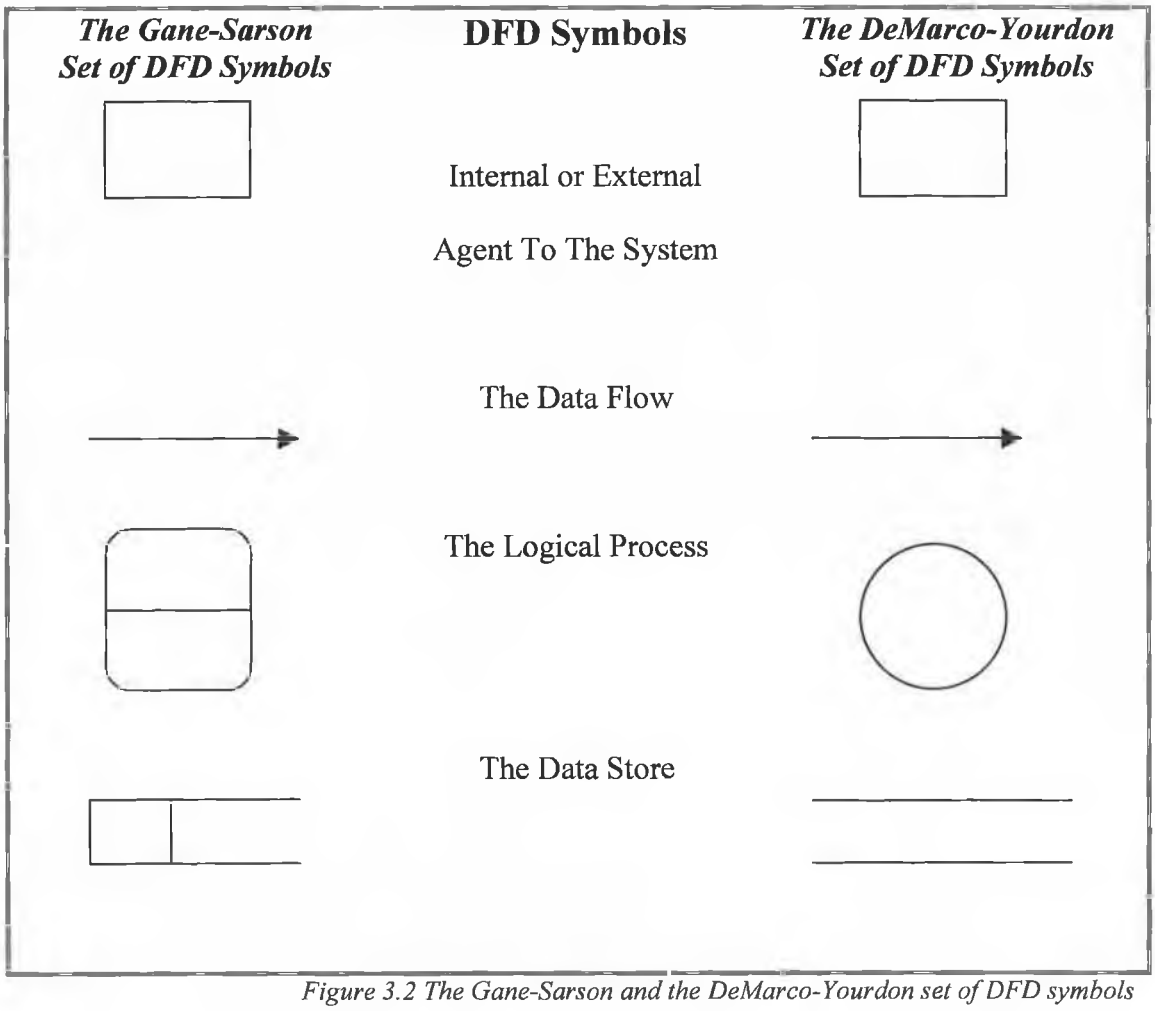


Figure 3.2 The Gane-Sarson and the DeMarco-Yourdon set of DFD symbols

Internal or External agents: These include people, organisations, or any other systems outside the system boundary that send data into the system or receive data from it. They are also known as sources or sinks of data. They are not part of the analysis.

Data Flow: These represent the movement of data between other components (Yeates el al., (1994). These are represented by an arrow and depict the fact that some data is flowing or moving from one process to another (Avison et al., 1995).

Processes: They represent activities in which data is manipulated by being stored or retrieved or transformed in some way (Yeates et al., 1994). They are shown on the Data Flow Diagram as rectangles. Each process has a unique reference number, in the top left-hand corner of the process box.

Data Stores: Data stores are used to represent stores of data within the system. They cannot be linked by data flows either to each other or to external entities.

3.4.1.3 Types Of Data Flow Diagram

Hoffer et al., (2002) identify four types of Data Flow Diagrams: *current physical model*, *current logical model*, *new logical model* and *new physical model.*

- *Current Physical Model:* In a current physical Data Flow Diagram, process labels include an identification of the technology used to process the data. Data flows and data stores are often labelled with the names of the actual physical media on which data flow or in which data are stored.
- *Current Logical Model:* The physical aspects of the system are removed as much as possible so that the current system is reduced to its essence, to the data and the processes that transform them, regardless of actual physical form.
- *New Logical Model*: The new logical model will differ from the current logical model by having additional functions, obsolete functions removed, and inefficient flows reorganised.
- *New Physical Model:* The new physical system represent the physical implementation of the new system.

The Data Flow Diagram (DFD) for the new system will reflect the decisions of the analysts about which systems functions, including those added in the new logical model, will be automated, and which will be manual.

3.4.1.4 The Data Dictionary

The data dictionary, initially one of DeMarco's techniques, is used to define the components in a Data Flow Diagram; processes, files, data flows and data elements. The data dictionary is the single authoritative source of detail about a project (Lejk et al, 2002). Data is normally hierarchically documented on pre-formatted forms in the data dictionary. See figure 3.3

Figure 3.3: The Hierarchy Of Data Dictionary Forms

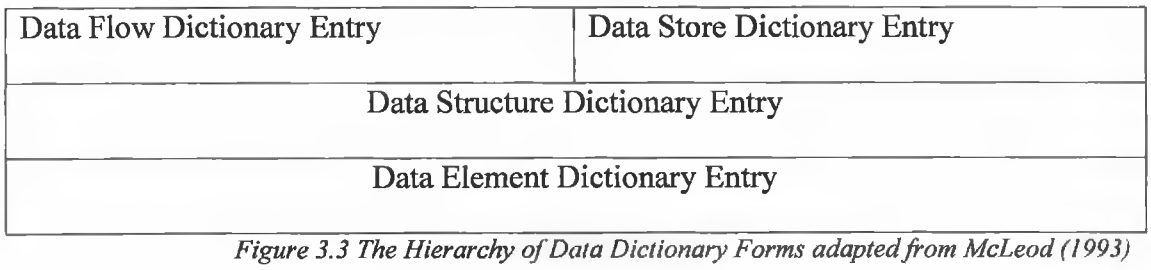


Figure 3.3 The Hierarchy of Data Dictionary Forms adapted from McLeod (1993)

3.4.1.5 Decision Table

Decision tables are used to express the conditional logic of processes where particular actions are resultant to different combinations of circumstances. Avison et al., (1995) provide the following definition of a decision table:

> *"Decision tables are tools which aim to facilitate the documentation of process logic, particularly where there are many decision alternatives."*

Hoffer et al., (2002) state that decision tables are a matrix representation of the logic of a decision, which specifies the possible conditions for the decision and the resulting actions.

The concept, use and format of decision tables are best described by way of example. Table 3.4, adapted from Whitten et al., (1998) depict the cheque cashing policy of a store. The upper section describes 'conditions' and the lower section of the table describes the 'actions' that are taken. Three conditions effect the decision to cash or refuse to cash a cheque. Firstly, the type of cheque, personal (P), company payroll (C); secondly, whether the cheque is drawn for €75.00 or less (Y/N); and thirdly, whether or not the company is accredited by LMART (Y/N).

Each of the four rule columns define a combination of conditions or rules, which result in the action specified by 'X' in the same column.

Conditions which are blank are irrelevant. For example, rule one prescribes that if a cheque is personal, and drawn for €75.00 or less, then company accreditation is irrelevant the cheque will be cashed.

Table 3.4: Decision Table Depicting A Cheque Cashing Policy

Conditions and Actions	**Rule 1**	**Rule 2**	**Rule 3**	**Rule 4**
C1: Type of cheque	Personal	Payroll	Personal	Payroll
C2: Check amount less than or equal to €75.00	Yes	Does not matter	No	Does not matter
C3: Company accredited by LMARK	Does not matter	Yes	Does not Matter	No
A1: Cash the check	X	X		
A2: Do not cash the check			X	X

Table 3.4 adapted from Whitten et al., (2001)

DeMarco (1979) discusses the advantages of using decision tables:

- They help document an understanding with the user.
- They help root out situations that have not been fully specified.

3.4.1.6 Structured English

The first use of the term "Structured English" may have been by Caine and Gordon, 1975. They described it as a means of solving communication problems between business clients and software developers. DeMarco (1979) states that the vocabulary of Structured English consists of: *Imperative English Language Verbs*, *Terms Defined In The Data Dictionary*, And *Certain Reserved Words For Logic Formulation.*

DeMarco (1979) provides the following definition of Structured English:

> *"Structured English is a specification language that makes use of a limited vocabulary and a limited syntax."*

The objective of structured English is to document process. Structured English used in conjunction with Data Flow Diagrams takes over where the Data Flow Diagrams leave off (McLeod, 1993). The overall structures of structured English specifications are built using the fundamental constructs that have governed structured programming. Valid control constructs include: *Sequence, Selection, and Iteration.* Following is a brief description of structured English in the context of *sequential instructions*, the *selection construct*, and the *iteration construct.*

3.4.1.6.1 Sequence

A *sequence construct* is a list of one or more actions or events. One or more crucial statements represent it. (Lejk et al., 2002).

Figure 3.4: The Sequence Construct

```
MULTIPLY PRICE BY QUANTITY-SOLD GIVING NET PRICE
MULTIPLY NET-PRICE BY 0.175 GIVING VAT
ADD VAT TO NET-PRICE GIVING GROSS-PRICE
```

Figure 3.4 The Sequence Construct – An Example

When documenting the Sequence construct, the entries are aligned on the same margin, one after another.

3.4.1.6.2 Selection

A *selection construct* occurs when there are a number of substitute policies that can apply, depending upon the result of some condition and only one policy is selected. (Lejk et al., 2002).

Figure 3.5: The Selection Construct

```
IF < INSERT CONDITION TESTED FOR> THEN;
        <INSERT INSTRUCTIONS>
        <INSERT INSTRUCTION>
```

Figure 3.5 The Selection Construct – An Example

3.4.1.6.3 Iteration

An iteration construct occurs when an action or series of actions are repeated subject to a condition that governs the continual repetition. (Lejk et al., 2002).

Figure 3.6: The Iteration Construct

```
ADD INOVICE-TOTAL TO OVERALL-TOTAL
 ADD 1 TO NO-INVOICE
UNTIL NO MORE INVOICES
DIVIDE OVERALL-TOTAL BY NO-INVOICE
GIVING AVERAGE-VALUE
```

Figure 3.6 The Iteration Construct – An Example

3.4.2 The Data Perspective

The data perspective places emphasis on a complete and thorough analysis of the data and its relationships (Olle et al., 1990). A knowledge and understanding of the nature of data is essential to the effective design of information needs. Lewis (1994) states that data analysis is included in most approaches of system development due to the fact that data analysis is a pre-requisite for efficient storage and organisation of data. Finkelstein (1989) states that information modelling involves the application of data analysis to a problem domain. The purpose is to gain an understanding of the problem domain by focusing on its data.

3.4.2.1 Information Modelling

Information modelling is also known as data modelling. Whitten et al., (1998) provide the following definition of data modelling:

> *"Data modelling is a technique for organising and documenting a systems data."*

Dowling (1998) states that '*a model is an abstract representation of something that exists in the real world.*' Information modelling therefore, is similar to modelling in general, the objective being to furnish an understandable representation of a problem. Data models are used to graphically represent the relationships between data required by a system.

Data models portray logical data structures, which illustrate the business requirements of a system. There are numerous conventions for data modelling, each emphasising contrasting concepts of data analysis. However, no model expresses all concepts of data analysis. Entities and relationships are universal to all models.

3.4.2.1.1 Entities

Hoffer et al., (2002) describes data entities as follows:

> *"An entity is a person, place, object, event or concept in the user environment about which the organisation wishes to maintain data."*

An entity is anything of importance about which information needs to be held (Dowling, 1998). An entity is an object or a group of objects. Peters (1988) states that an object is a *"thing"* (e.g., person, place, organisation, function) about which the organisation wishes to collect information. Shlaer et al., (1988) provide the following definition of an object:

> *"An object is an abstraction of a set or real world things."*

Entities are shown in a box and are descriptively named in uppercase using a singular noun or noun phrase. Figure 3.7 depicts two entities, EMPLOYEE and DEPARTMENT, and the relationship between them (WORKS IN).

Figure 3.7: Entity And The Relationship Describing Business Relevance

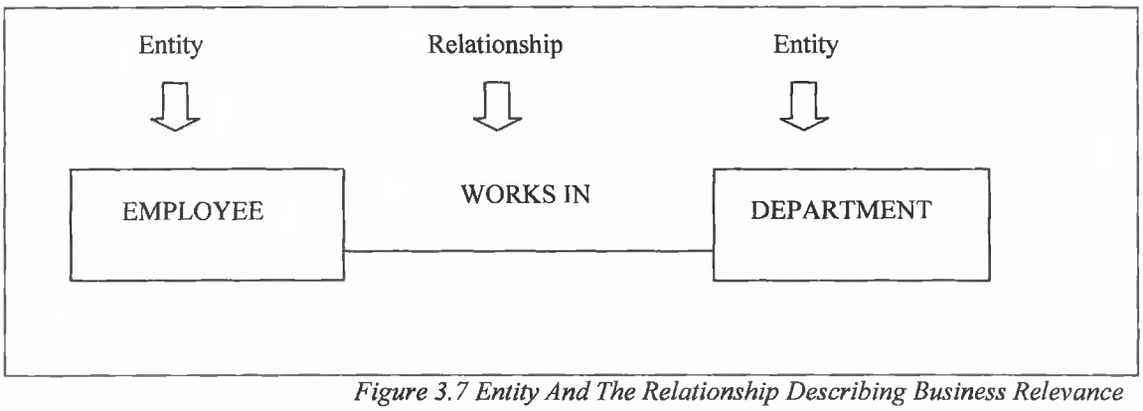


Figure 3.7 Entity And The Relationship Describing Business Relevance

An attribute is a data element associated with an entity. Shlaer et al., (1988) define an attribute as follows:

> *"An attribute is the abstraction of a single characteristic possessed by all the entities that were, themselves, abstracted as a object."*

Attributes are used to identify, classify and describe the state of an entity. Each entity has a set of attributes that are associated with it. For example EMPLOYEE NUMBER, LAST NAME and FIRST NAME are attributes of the entity STAFF MEMBER, and DEPARMENT NUMBER is an entity of the entity DEPARTMENT.

3.4.2.1.2 Relationship

Benyon (1997) provide the following definition of a relationship:

> *"A relationship is an association or link between two or more entities."*

McFadden et al., (1999) defines relationship as follows:

> *"A relationship is an association among instances of one or more entity types that is of interest to the organisation."*

A relationship is an association between two entities that is important to the system. Relationships are important when designing information systems because they define access from one entity occurrence to another (Ashworth, 1990). There are three types of relationships, one to one (1:1), one to many or many to one (1:M), and many to many (M:M) which describes the possibility that entities may have many associations in either direction. (Kendall et al., 1999). See Figure 3.8.

Figure 3.8: One-To-One, One-To-Many, Many-To-Many Relationships

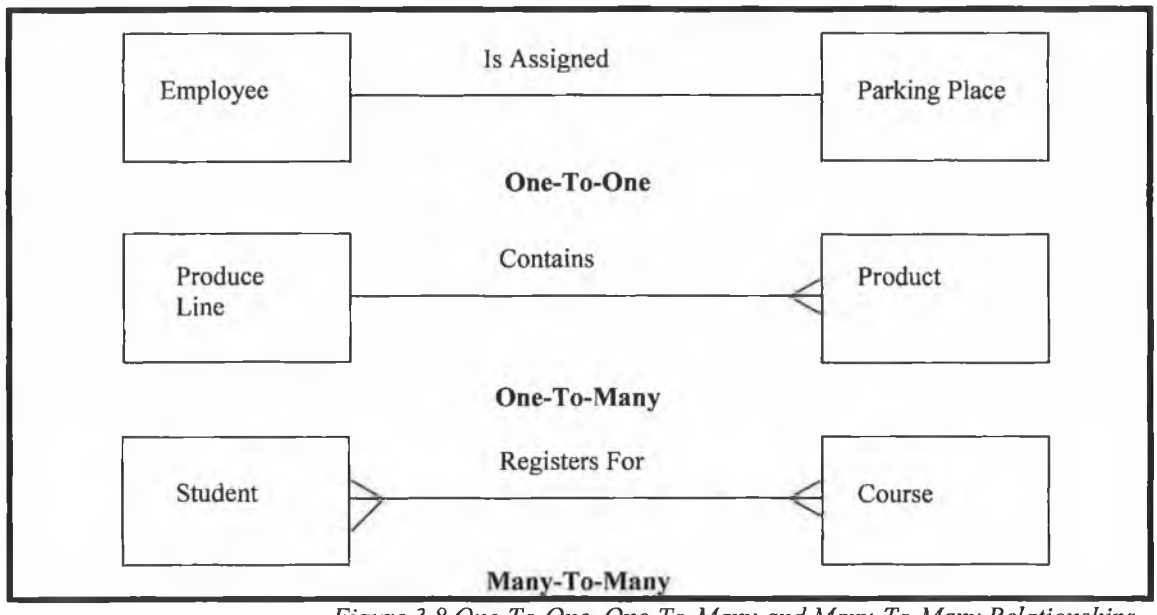


Figure 3.8 One-To-One, One-To-Many and Many-To-Many Relationships

3.4.2.1.3 The Data Modelling Process

Entity Relationship Diagrams are used to document the organisations data by identifying the data entities and showing the relationship that exists between them. System analysts and end users develop them. The Entity-Relationship model is highly flexible and powerful for exploring information needs.

McLeod (1993) identifies seven steps in the process of developing Entity-Relationship diagrams.

Step 1: Identifying the entities within the problem domain e.g. STAFF MEMBER and DEPARTMENT.

Step 2: Defining the relationship that occurs between the entities, e.g. STAFF MEMBER and DEPARTMENT.

Step 3: Preparing a rough Entity Relationship Diagram.

Step 4: Mapping data elements to the entities.

Step 5: Performing data analysis. The purpose is to develop data structures which can be transformed into efficient databases. Changes in data structures can severely test system integrity and program logic. Efficient databases can be achieved by applying normalisation (developed by Codd, 1970) to the data. Normalisation is a tool to validate and improve logical design.

Dowling (1998) states that normalisation ensures that the entities that are produced during data modelling are designed in the best possible way for modelling the information requirements of a system. During the process of normalisation it is imperative to identify key entity attributes as they are exclusive to individual entities and can be used as identifiers. Normalisation is completed in three stages (McFadden, 1999);

- First Normal Form: This is achieved by removing repeating groups.
- Second Normal Form: This is achieved by ensuring that all data items are dependent on the key, that is, any partial functional dependencies have been removed.
- Third Normal Form: Ensuring that all other items are independent of each other, that is, any transitive dependencies have been removed.

Table 3.5: Event List	
EVENT NUMBER	**EVENT DESCRIPTION**
1	Cashier switches pump on.
2	Cashier switches pump off.
3	Customer lifts petrol gun.
4	Cashier requests to clear register.
5	Customer replaces gun.
6	Customer selects blend.
7	Dispenser reports unit dispensed.

Table 3.5 Example of an Event List adopted from Allen (1991)

3.4.3.1 Types Of Events

Satzinger et al., (2002) identifies three types of events; *External Events, Temporal Events and State Events.*

An *external event* is an event that occurs outside the system, usually initiated by an external agent. A typical example of an external agent is that of a customer who may want to place an order for one or more products.

A *temporal event* is an event that occurs as a result of reaching a point in time. An example of a temporal event would be that of a payroll system, as the information system produces outputs at defined intervals.

A *state event* is an event that occurs when something happens inside the system that triggers the need for processing. For example if the sale of a product results in an adjustment to an inventory record and the inventory in stock drops below a reorder point, it is necessary to reorder.

3.4.3.2 The Entity Life History

Ashworth et al., (1990) describes the entity life history as follows:

> *"An entity life history is a diagrammatic representation of the life of a single entity, from its creation to its deletion. The life is expressed as the permitted sequence of events that can cause the entity to change."*

Entity life histories examine the effects, which events have on entities. They describe the events of the life on an entity. They are the link between process models and data models and are a major part of structured methodologies (Avison et al, 1998). The entity life history borrows its symbols and some of its techniques from Jackson Structured Programming (JSP) (Green, 1996).

3.4.3.3 The Entity Life History and Control Constructs

Entity life history diagrams use the following diagramming components: *Sequence, Selection,* and *Iteration.* A sequence is represented by a series of boxes reading from left to right.

A selection is represented by a set of boxes with circles in the top right corners, thus each transaction within the iteration of transactions can be a pay deposit, a direct deposit or a cheque cashing transaction. An asterisk in the top right-hand corner of a box represents iteration. See figure 3.10.

Figure 3.9: Entity Life History

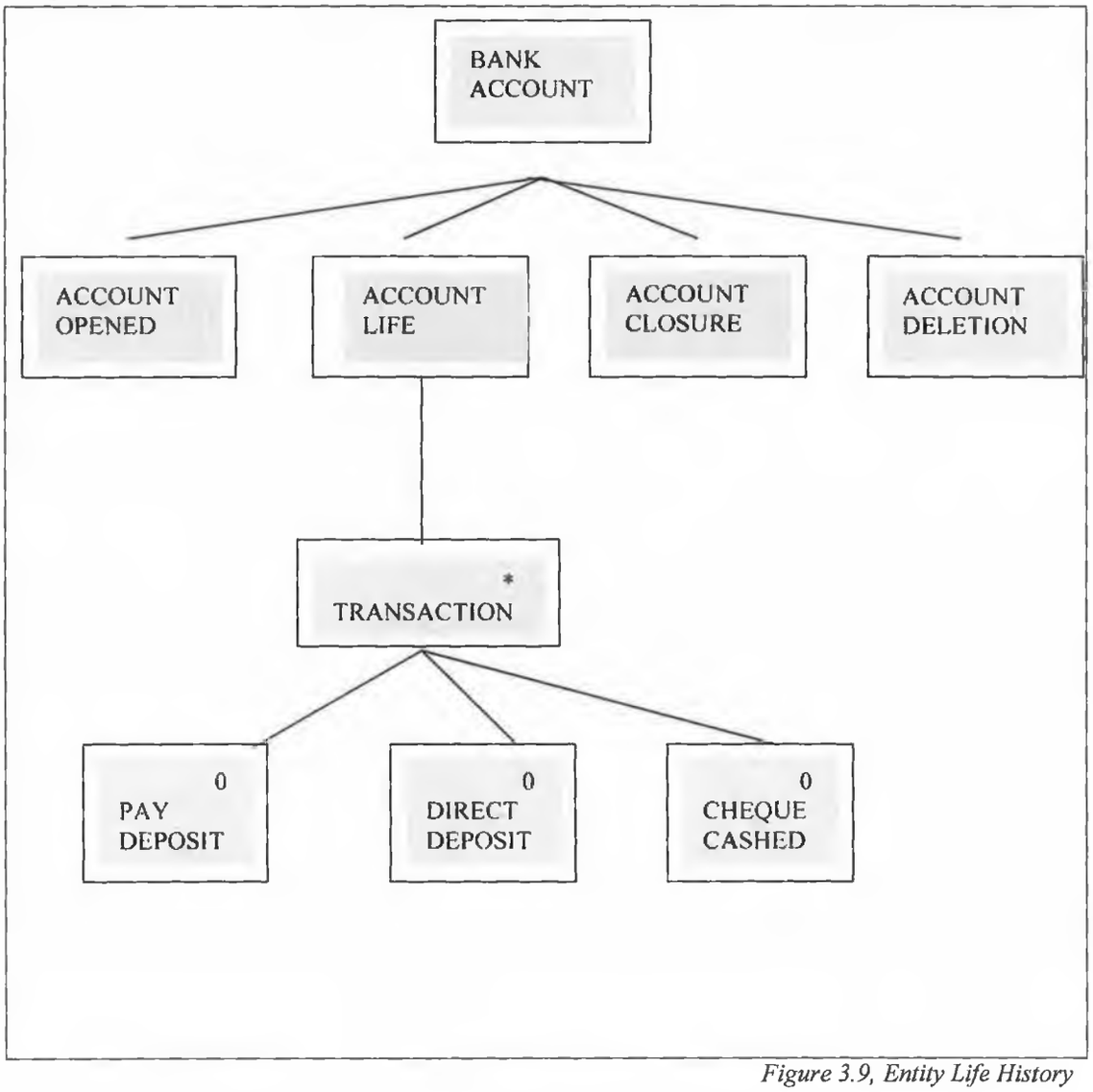


Figure 3.9, Entity Life History

Figure 3.9 depicts the life of an entity BANK ACCOUNT in terms of permitted events that cause it to change, demonstrating this modelling technique.

3.5 Information System Development Methodologies

Whitten et al., (1998) defines a methodology as follows:

> *"The physical implementation of the logical life cycle that incorporates (1) step-by-step activities for each phase, (2) individual and group roles to be played in each activity, (3) deliverables and quality standards for each activity, and (4) tools and techniques to be used for each activity."*

Traditional methodologies focus on the development process, which generally consists of six steps. A generic methodology may consist of planning, gathering requirements, analysis, design, coding, testing, implementation review, and maintenance (Trepper, 2000). Methodologies can differ according to each particular project. Methodologies used in traditional systems development may differ from the methodologies used in Web application development. The use of formalised methodologies for Web application development will be discussed later in chapter 4.

Norman (1996), defines the purpose of a methodology as follows:

> *"The purpose of a methodology is to promote problem-solving strategy by pre-selecting the methods and techniques to be used."*

Methodologies have a tendency to evolve over time. During the 1960's traditional systems analysis and design was dominant, in the 1970's structured analysis and design was the dominant methodology, in the 1980's information engineering became apparent, and today object-oriented analysis and design is a dominant methodology.

3.5.1 Modelling Techniques And Methodologies

Techniques are non-proprietary and tend to be incorporated into many different methods (Russo et al., 2002). Particular techniques may feature in a number of methodologies. For example Data Flow Diagrams (DFD's) are used by the Structured Systems Analysis and Design Methodology (SSADM), Entity Relationship Diagrams are used by the Information Engineering (IE) methodology.

3.6 Information Engineering

Clive Finkelstein and James Martin developed this approach in 1981. Information engineering is a comprehensive methodology, which covers every aspect of the development life cycle. It is a framework in which a diversity of techniques is used to develop information systems in a proficient manner (Avison et al., 1995). The concept behind Information Engineering is that information systems should be engineered like other products.

Whitten et al., (2001) describes Information Engineering as follows:

> *"a data-centred, but process-sensitive technique that is applied to the organisation as a whole."*

Information Engineering is deeply rooted in data analysis, which is based on detailed analysis of the variety of documentation used within an organisation. Finkelstein (1993) quotes his earlier work Finkelstein (1989) to define Information Engineering as follows:

> *"Information Engineering is an integrated set of techniques, based on corporate strategic planning, which results in the analysis, design and development of systems which support those plans exactly. Information Engineering is applied by managers and users with no knowledge of computers, but instead with an expert knowledge of their business - in conjunction with expert system which provide rapid feedback to management for refinement of the strategic plans."*

Kerr (1991) defines Information Engineering as follows:

> *"a family of data-driven analysis and design techniques that support the logical design phase of the system development life cycle."*

Information Engineering is a methodology for developing integrated information systems based on the sharing of common data. It differs from the more traditional system development methodologies in that it emphasises the need to integrate information and its relationships with functional processes.

3.6.1 The Stages Of Information Engineering

Information Engineering consists of three main stages; an analysis phase, a design phase and a generation phase (Finkelstein, 1993).

Table 3.6: Phases Of Information Engineering

Analysis Phase
Project Scope Stage
Strategic Modelling Stage
Strategic Modelling
Strategic Objective Modelling
Strategic Refinement
Tactical Modelling Stage
Tactical Modelling
Tactical Objectives Modelling
Tactical Refinement
Operational Modelling Phase
Strategic Design Stage
Tactical and Operations Design Stage
Design Phase
Strategic Design Stage
Tactical and Operations Design Stage
Generation Phase
Implementation Strategic Stage
Systems Generation Stages

Table 3.6 Phases of Information Engineering (Russo et al., 2002).

3.6.1.1 The Analysis Phase

The analysis phase uses strategic and tactical statements to identify data needed to support the organisation. The progression through the analysis phase involves four stages; *project scope, strategic modelling, tactical modelling*, and *operational modelling.*

3.6.1.1.1 The Project Scope Stage

The project scope stage establishes the objectives and the boundaries of a project. Project aims, deliverables, and completion time are established. Strategic directions for the project area, based on strategic plans, or on statements of direction are identified. Software tools to be used for strategic and tactical modelling stages are chosen. Based on the project size, the budget is set.

3.6.1.1.2 The Strategic Modelling Stage

This stage produces the strategic model. The strategic modelling stage is further subdivided into three steps; strategic modelling, strategic objectives modelling and strategic refinement. Strategic modelling develops a strategic model of entities and associations that represent information of interest to senior management of the project area. The strategic model may also contain strategic attributes for achievement of strategic goals.

Strategic objective modelling identifies criteria for performance modelling, and defines strategic data required to measure performance. The strategic model is refined in the strategic refinement phase.

3.6.1.1.3 Tactical Modelling Stage

The tactical modelling stage is used to further define the strategic model by producing a tactical data model appropriate to key functional areas and relevant to middle management. Tactical modelling develops operational models that contain information of interest to middle management of the project area. These may contain non-derived attributes, or derived attributes from more detailed operational data. The relevant data is identified in detail for each functional area. The data required to measure the achievement of tactical objectives on this level are identified.

3.6.1.1.4 The Operations Modelling Stage

The operations modelling stage focuses on the day to day operations of the organisation. Operational modelling develops operational models of fundamental data that already exist in current manual or automated systems, or may be required for new directions defined in strategic and tactical models. This stage is performed in three steps: current system modelling, operations objectives modelling and operational refinement.

3.6.1.2 The Design Phase

This phase comprises strategic design, tactical design and operational design. During strategic design data is checked for completeness and consistency of definition. Where several strategic models exist they are amalgamated to produce one integrated data model. Analogous to strategic design, integrated tactical and operational data models are also produced.

3.6.1.3 The Generation Phase

In this phase the data models are used to generate database systems. There is no specific assistance provided to determine alternate strategies. Techniques from other methods may be imported for deciding on the different types of systems that might be implemented. Selected parts of the data models are used as a basis for the generation of definition statements for database systems.

3.6.2 Strengths Of Information Engineering

Finkelstein (1989) identifies several benefits of Information Engineering. These are summarised below:

- It is data oriented, identifying data vital to the organisation and supportive of management in their capacity as decision makers.
- It is objectives driven.
- Information Engineering can be driven by high level strategic objectives and proceeds in a top down fashion.

- It draws on the expert knowledge of users, using them to identify vital data and produce models.
- It achieves data consolidation, producing an integrated data model useable throughout the organisation.

3.6.3 Weaknesses Of Information Engineering

The greatest problem of Information Engineering is the initial implementation, which is likely to be expensive and will not yield immediate financial results (Yeates et al., 1994). Information Engineering is dependant on CASE Tools and the ability to reverse engineer is predicted on the beliefs on an integrated CASE tool with a sophisticated repository (Avison et al., 1995)

3.7 Structured Systems Analysis And Design Method (SSADM)

Structured Systems Analysis and Design Method (SSADM) is not generally recognised as being part of a structured approach (Russo et al., 2002). It is a data-driven approach. This means that there is a basic assumption that the systems have an underlying data structure, which can change very little over time. The underlying data structures are modelled from an early stage. Structured Systems Analysis and Design Method (SSADM) consists of three important features: *structures, techniques* and *documentation.*

The structures of the Structured Systems Analysis and Design Method (SSADM) identify that each stage is broken down into a number of steps. These steps define inputs, outputs and tasks that need to be performed. The products of each step and the interfaces between the steps are defined. The structured techniques give standards for how each step and task is to be performed. Documentation defines how the products of a development activity should be presented. Several different types of documents will be produced during a project: *diagrams, forms, matrices*, and *narrative reports*.

Structured Systems Analysis and Design Method (SSADM) takes three basic views of an information system: *logical data structure* which identifies what information is stored and how it is interrelated, *Data Flow Diagram,* which identifies how information is passed around, and *entity life histories*, which identifies how information is changed during its lifetime.

Structured Systems Analysis and Design Method (SSADM) comprises of a hierarchy of modules, stages, steps and tasks, which serve to divide the development process into a number of phases. The Structured Systems Analysis and Design Method (SSADM) modules are: feasibility study, requirements analysis, requirements specification, logical systems specification and physical design, see Table 3.7.

Table 3.7: Steps Of SSADM Method

Module 1: Feasibility Study (FS)
Stage 0: Feasibility
Steps:
010 Prepare for Feasibility Study
020 Define the Problem
030 Select Feasibility Option
040 Assemble Feasibility Report
Module 2: Requirements Analysis (RA)
Stage 1: Investigation of Current Environment
Steps:
110 Establish Analysis Framework
120 Investigate and Define Requirements
130 Investigate Current System
140 Investigate Current Data
150 Derive Logical View of Current Services
160 Assemble Investigation Results
Stage 2: Business System Options
Steps:
210 Define Business System Options
220 Select Business System Option
Module 2: Requirements Specification (RS)
Stage 3: Definition of Requirements
Steps:
310 Define Required System Processing
320 Develop Required Data Model
330 Derive System Function
340 Enhance Required Data Model
350 Develop Specification Prototypes
360 Develop Processing Specification
370 Confirm System Objectives
380 Assemble Requirements Specification
Module 4: Logical System Specification (LS)
Stage 4: Technical System Options
Steps:
410 Define Technical System Options
420 Select Technical System Options
Stage 5: Logical Design
Steps:
510 Define User Dialogues
520 Define Update Processes
530 Define Enquiry Processes
540 Assemble Logical Design
Module 5: Physical Design (PS)
Stage 6: Physical Design
Steps:
610 Prepare for Physical Design
620 Create Physical Data Design
630 Create Function Component Implementation Map
640 Optimise Physical Data Design
650 Complete Function Specification
660 Consolidate Process Data Interface
370 Assemble Physical Design

Table 3.7 Steps of SSADM Method adapted from Russo et al., (2002).

3.7.1 Feasibility

This stage is concerned with ensuring that the project in the planning phase is feasible, ensuring that it is technically and financially beneficial. This phase has four steps: *prepare for the study*, which assesses the scope of the project; *define the problem*, which compares the requirements with the current position, *select feasibility option*, which considers alternatives and selects one, and assemble *feasibility report.* (Avison et al., 1998).

3.7.2 Investigation Of Current Environment

Requirement analysis consists of two stages: *investigation of the current requirements* and *business system options*. The first of these stages reiterates much of the work carried out at the feasibility study stage in greater detail however. The results of the feasibility study are examined and the scope of the project reassessed and the overall plan agreed with management. The requirements of the new system are examined along with investigating the current processing methods and data of the current system again in greater detail (Avison et al., 1998).

3.7.3 Business System Options

This stage determines the functionality of the new system. A number of business system options are outlined, all satisfying the minimum set of user requirements, and a few of these are presented to management so that one can be chosen.

Each of these will have an outline of the cost, development time scale, technical constraints, physical organisation, volumes, benefits and impact on the organisation. The option chosen is documented in detail and agreed as the basis of the new system. (Avison et al., 1998).

3.7.4 Definition Of Requirements

This stage leads to the full requirements specifications and provides clear guidance to the design stages that follow. The emphasis is on determining the desired system data, functions and events. Prototyping techniques are also suggested for the development of human-computer interface.

3.7.5 Technical System Options And Logical Design

This stage assesses the different options for implementing a part of the specification and describes options, costs, benefits and constraints. External constraints consist of time, cost, business performance and any hardware or software restrictions set in the feasibility study. Internal constraints are (a) responsiveness – the responsiveness of a system is decided, (b) sizing of files (c) security and (d) interfacing to other systems. The aim is to select the best set of technical products that meet the requirements. A technical environment description for the chosen options is input to logical design. Logical database design process uses information from entity-event modelling to construct update process models. Only non-procedural specifications are produced in the logical design stage (Flynn, 1992).

3.7.6 Physical Design

The logical design is converted to a design that fits the computer hardware and software selected. Physical design involves the specification of files, the specification of programs, and the operating and manual procedures that support them. (Ashworth et al., 1990).

3.8 Strengths Of Structured Systems Analysis And Design

Walters et al., (1994) note that the step-by-step procedure, from a current physical view of the system to the required physical view, is easier for users to comprehend rather than large conceptual leaps. Fitzgerald (2000) identifies that Structured Systems Analysis and Design Method (SSADM) originates from the MODUS methodology, which was dominant between 1965 and 1997. This is an advantage as the methodology is mature and well proven.

3.9 Weaknesses Of Structured Systems Analysis And Design

Walters et al., (1994) identifies the following weaknesses of Structured Systems Analysis and Design Method (SSADM).

- Certain areas of the system may be looked at in isolation, this then resulting in any knock-on effects being missed or ignored.

- Structured approaches inherit the problems of the System Development Life Cycle (SDLC) on which they are based.
- Rigid framework may create difficulties in soft problem situations, where it may be necessary to explore areas in the problem situation not catered for in the framework.

3.10 Object-Orientation Development: An Introduction

Object-Orientation invaded the software development process as a programming construct in the SIMULA language. It was suited to simulating real world objects as software objects, due to its potential in modelling semantic relationships and associations among objects.

Object-Oriented Programming emerged as a term associated with the development of Smalltalk, the original object-oriented language, and the one most often regarded as a pure object-oriented language (Loy, 1989). Smalltalk was developed in the early seventies and it was designed to completely envelop the object-oriented approach.

The programming language C++, developed in the mid eighties, is a hybrid object-orientated language and has featured prominently in the history of object-orientation. In contrast to Smalltalk, considered to be a pure object-oriented language, C++ offers object-oriented features in a traditional language.

Java took the software world by storm due to its close ties with the Internet and Web browsers. Java offers great promise as the standard Internet and Intranet programming language (Montlick, 2003). More recently the advent of Visual.Net makes the embracing of Object-Oriented methods easier by the use of Microsoft's popular visually based software development environment.

3.11 An Introduction To Objects And The Object-Oriented Paradigm

The Object-Oriented (OO) paradigm, takes the same components of a software system: *data* and *procedures*, however, it de-emphasises the procedures, stressing instead the encapsulation of data and procedural features together, exemplified by the clear and concise specification of the module interface. In system decomposition, based on an object-oriented approach, the system is viewed as a collection of objects, sometimes referred to as entities or object classes (Henderson-Sellers et al., 1990). The transition to object-oriented techniques involves adapting a new unit of modularity (Cox, 1987).

Object-oriented systems are comprised of objects. An object can be a tangible entity such as a person or an event. Objects have properties that can be used to describe them, such as a state, and a relationship to other objects. They behave in a certain way in response to a given situation.

An object has both data (state) and functional (behavioural) components. Objects encapsulate both *data* and the *functions* that manipulate the data.

Whitten et al., (1998) define objects as follows:

> *"Something that is or is capable of being seen, touched, or otherwise sensed, and about which users store data about associate behaviour."*

A class is a template for an object. A class encapsulates the data and procedural abstractions that are required to describe the content and behaviour of real world entities. Object-oriented analysis and design is primarily concerned with the identification and specification of classes: their structure and behaviour, their attributes and relationships with other classes, their function or the service they provide, and their dynamic behaviour.

3.12 The Concepts

Object-orientation is a technique for system modelling. Using object-orientation it is possible to model the system as a number of concepts that interact with one another (Jacobson et al., 1993). Object-orientation looks at a domain that is of interest, and then identifies the key abstraction and relationships. For every entity in that domain, there is an object that represents that concept in the model (Carmichael, 1994).

It is generally agreed that object-orientation includes three essential concepts: *Notation of encapsulation* and/or *information hiding, abstraction* by classification, and *polymorphism* as implemented through inheritance (Henderson-Sellers, 1992).

Norman (1996) discusses the characteristics of object-orientation: *Message communication, Encapsulation, Abstraction, Polymorphism, Inheritance and Class reusability*

3.12.1 Message Communication

Objects interact by means of messages. Messages contain a name that identifies its destination, and may also contain some arguments or parameters. In an object-oriented language, messages from one object to another can involve a lot of processing because it is not always possible to assume that the sender knows the address of the receiver. This is referred to a dynamic binding.

3.12.2 Encapsulation

Central to object-oriented development is the idea that it is possible to model the problem domain as an arrangement of reasonably autonomous, encapsulated objects that access each other through external and protected interfaces.

Norman (1996) defines encapsulation as:

> *"the notion that a software component (module, subroutine, method, and so on) should isolate or hide a single design decision."*

Encapsulation refers to the inclusion of everything an object needs, and doing it in such a way that no other object need ever be aware of this internal structure. In systems analysis and design, system analysts decompose the problem domain into encapsulated units.

Encapsulation helps to localise their volatility when changes in maintenance are required. Information hiding refers to any mechanism that allows certain portions of the information system to be hidden.

3.12.3 Abstraction

Abstraction is the principle of ignoring those aspects of a problem domain that are not relevant to the current purpose. Therefore system analysts must choose certain things over others. Abstraction is important for the management of complexity.

Graham (1993) defines abstraction as follows:

> *"Representing the essential features of something without including background or inessential detail."*

In programming terms, this means that objects should abstract and encapsulate both data and processes.

Coad et al., (1991b) discuss two forms of abstraction known as: *procedural abstraction*, which is used extensively by requirements analysts and *data abstraction*, used to specify the systems responsibilities.

3.12.4 Polymorphism

Polymorphism is the ability to send the same message to different objects and each object responding in its own specific manner. In a general sense, it is the ability to take on different forms. Polymorphism facilitates the reuse of code in different instances. There are three classifications of polymorphism: *inclusion, operation*, and *parametric* (Wilkie, 1994).

Brown, (2002) states the following with regard to polymorphism;

- Polymorphism means, "occurring in various forms."
- Polymorphism means a behaviour may be inherited unchanged, or it may be totally different between the parent and child classes; and, in the same way as for attributes, it allows us to specialise a behaviour for particular subclasses.
- Polymorphism is possible only in object-oriented environments and is part of the definition of object-oriented.

- Abstract classes, which have no direct instances, are often introduced to exploit inheritance and polymorphism.

3.12.5 Inheritance

Inheritance is a key concept and a definitive concept in object-orientation. It provides a method of relating classes in a way that is semantically sound. It is the only way that classes relate to each other. Wilkie (1994) states "*Inheritance provides a mechanism for managing classes and for sharing commonalty*".

Coad et al (1991b) define Inheritance as follows:

> *"A mechanism for expressing similarity among Classes, simplifying definition of Classes similar to one(s) previously defined. It portrays generalisation and specialisation, making common Attributes and Services explicit within a Class hierarchy."*

Subclasses inherit attributes and methods from their super classes. For example, Managers, Sales Persons and Engineers inherit characteristics from the Employee Class. They all have an employee number, a name, address, date when current employment commenced. All of these attributes could be inherited from the Employee Class. This type of inheritance is called single inheritance because each of the subclasses inherits from no more than one super class.

However, a second type of inheritance, Multiple Inheritance occurs when a class inherits from more than one immediate super class.

3.12.6 Class Reusability

Reuse in the context of software development effort involves the *"using again"* of components which are a product of the domain which encompasses systems analysis, systems design and system construction. To practitioners it involves making use of existing knowledge and artifacts throughout the entire life cycle. Reusability can be justified both economically because of increased productivity and intellectually because it simplifies the understanding of the phenomena (Wilkie, 1994). Software reusability has many different forms, each with a different economic payoff.

Wegner (1990) identifies four discrete kinds of reuse;

- *Inter-application reusability:* The reuse of software components in a variety of applications.
- *Development reusability:* Involves the reuse of components in successive versions for a given program.
- *Program reusability:* Involves the reuse of programs in successive executions with different data.

- *Code reusability:* Involves the reuse of code during the single execution of a program. This form of reusability benefits from modularity, where a module performs some pre-described function and is called wherever this functionality is required.

3.13 Object-Oriented Development And The Application Life Cycle

The Object-Oriented Development Life Cycle (OODLC) is the traditional systems development life cycle in a modern form, however stressing the use of objects. The models used in the object-oriented paradigm, are different to those used in the traditional systems development (Brown, 2002). An object-oriented software development life cycle should be more incremental and iterative in nature compared with traditional structured life cycles (Wilkie, 1994).

The analysis phase covers from the initiation of the project, through to users needs and feasibility study. The design phase covers the various concepts of system design, logical design, detailed design, program design and physical design. Following from the design stages, the computer program is written, the program tested, in terms of verification, validation and sensitivity testing, and when found acceptable, put into use and then maintained well into the future. See table 3.8.

Table 3.8: Object-Oriented Development Phases

Phase	Activity	Models Produced	Components
Analysis	OOA	Requirements Model	Project Scope Feasibility Study Context Diagram
		Object Model	Class Diagram: Entity Classes Interface Classes Control Classes Behaviour Diagrams: State chart Diagrams Collaborations and CRC Cards Sequence Diagrams Activity Diagrams
Design	OOD	Design Versions of the OO Models	
Construction	OOP	Actual System	
Testing	O-O Testing	Working System	
Maintenance	All of the above	All of the above	

Table 3.8 Object-Oriented Development Phases Adopted by Brown (2002)

3.14 Object-Oriented Development Methodologies

The following section presents a summary of the major object-oriented development methodologies. This is intended to highlight specific differences and their corresponding advantages and disadvantages.

3.14.1 Shlaer and Mellor: Object-Oriented Systems Analysis

This method (Shlaer and Mellor, 1988) is an adaptation of traditional structured methods using entity modelling.

Initially Object-Oriented Systems Analysis by Shlaer and Mellor (1988) was essentially information analysis, however, it failed to capture behaviour and did not contain inheritance or classification, therefore, it was not completely object-oriented. Shlaer and Mellor further developed this method, so that it can deal with behavioural properties of objects (Wilkie, 1994).

The method breaks down into three steps; *the information model, the state model* and *the process model.*

3.14.1.1 The Information Model

The cornerstone of the method is the information model. Four tools are presented for achieving the information model resultant to the analysis process. Two of these are in graphical form and based on the Entity Relationship Diagram. The first, the information structure diagram is used to declare the objects, the attributes, and the relationships between the objects. The second, the overview information structure diagram is similar but does not contain attribute names. The purpose of it is to present an uncluttered view of the problem domain. The model is used to identify the entities in the problem domain. The information model formalises knowledge about the world in terms of objects, attributes and relationships. It is a data-oriented view of the system.

3.14.1.2 The State Model and Process Model

On completion of the information model Shlaer and Mellor recommend the development of a state model. This is achieved by building state transition diagrams for objects that exhibit a life cycle. State transition diagrams instigate actions that are represented as actions on a Data Flow Diagram (process model).

The information model, state model, and the process model are integrated as follows: objects in the information model become stores in the process model, all data for processing or resulting from processes and defined in the information model. Process may cause events specified on the state model.

Shlaer and Mellor analysis produce the following outputs:

Information Modelling Stage

- Information structure diagrams. These are essentially traditional Entity Relationship Diagrams.
- Object and attribute descriptions.
- Relationship descriptions.

State Modelling Stage:

- State transition diagrams.
- State transition tables.
- A description of each action on the state transition diagram.

Process Modelling Stage

- Data Flow Diagrams to represent the internal transformations associated with an action on the state transition diagrams.

Object-Oriented Systems Analysis ignores the encapsulation of services and only assists the inheritance of attributes. Services or methods, while not inherited abilities, are curtailed to the level of function with regard to functional decomposition or process with regard to Data Flow Diagrams. Objects as defined in Object Oriented System Analysis are not polymorphic and are stripped of the ability to react alternatively to the same message or interface. Coad et al., (1991b) present a method and model, which overcomes these shortcomings and is purer in the object-oriented tradition, object-oriented analysis (Coad et al., 1991b).

3.14.2 Coad And Yourdon: Object-Oriented Analysis

Object-Oriented Analysis (OOA) is a data driven approach modelled with the entity-relationship diagram. Coad and Yourdon (1991b) identify five major layers: *the class and object layer*, *the structure layer*, *the subject layer, the attribute layer, and the service layer.* They are briefly described below.

3.14.2.1 The Class And Object Layer

This layer is concerned with finding suitable classes to describe the problem domain. The method suggests ways to identify classes such as looking at structures, external systems, devices, events remembered, roles played, operational procedures, organisational units and physical locations. The output from this stage is a set of classes with names but little else. These diagrams are called *class* and *object diagrams.*

3.14.2.2 The Structure Layer

The output from this stage is a set of inheritance and whole-part relationships between the classes/objects identified previously, with the possible addition of classes identified as a result of considering the structure. The results from this stage are generalisation-specialisation structure diagrams and whole part structure diagrams.

3.14.2.3 The Subject Layer

A subject is a means for controlling how much of a model a reader considers at one time. It is a way of segmenting a problem, creating a boundary between groups of closely related classes. Subject diagram are created at this stage.

3.14.2.4 The Attribute Layer

An attribute is a data element used to describe an instance of a class. At this stage the method adds instance connections - representing associative relationships (with cardinality) between classes or objects. This stage gives rise to attributes diagrams and instance connection diagrams.

3.14.2.5 The Service Layer

A service is the processing to be performed upon receipt of a message. This is the first point at which the dynamics of the environment being analysed are considered. The method introduces the object lifetime concept of state. The results from this stage are service diagrams, message connection diagrams, service/state tables, and object state diagrams.

Carmichael (1994) states that the greatest weakness of Object-Oriented Analysis is its simplicity. Object-Oriented Analysis and Object-Oriented Design fit together in a consistent, systematic approach to more effective analysis and design.

3.14.3 Coad And Yourdon: Object-Oriented Design

Object oriented design addresses the application and the infrastructure for the application and focuses on the representation of four major system components (Pressman, 2000).

Object Oriented Design adds four design criteria to the analysis model: *Problem Domain Component, Human Interaction Component, Task Management Component* and *Data Management Component.* Wilkie (1994) discusses the following:

3.14.3.1 Problem Domain Component

This component takes the results of the analysis phase and adds detail to the identified classes to make them implementable such as including existing classes from commercially available libraries. The method describes ways to design reusability into the classes brought from the analysis phase.

3.14.3.2 Human Interaction Component

This component is concerned with the design of the user interface. It introduces additional objects required to create a user interface such as presentation and dialogue objects.

3.14.3.3 Task Management Component

This component is concerned with identifying the tasks in the system. Architectural issues are considered at this stage. Design considerations include threads of control within an application such as concurrency and task management across several processors. The scheduling and intercommunication of tasks are also part of this component.

3.14.3.4 Data Management Component

This component is concerned with the data aspects of the system, such as how data is stored and retrieved and whether flat-file, relational or object-oriented schema should be used. The issue of normalised data, 1^{st}-5^{th} normal form to remove data redundancy is considered here.

Object-Oriented Design results in a design that achieves a number of different levels of modularity. Major system components are organised into subsystems. Data and the operations that manipulate the data are encapsulated into objects. Object-oriented design builds upon four important software design concepts: *abstraction, information hiding, functional independence* and *modularity*. Object-Oriented Design enables the designer to achieve all four without complexity or compromise (Pressman, 2000).

Wilkie (1994) argues that Object-Oriented Design does not prescribe a development life cycle, which if in place would aid software development managers.

3.15 The Benefits Of Object-Oriented Techniques

As with any new technique, object-oriented development will survive beyond the initial adoption only if it delivers the benefits as promised. Brown (2002) identifies the following benefits of object-oriented techniques. These are summarised below:

System Stability: Object oriented techniques are resilient to change. Changes can be made to a system with minimal time and effort.

Maintainability: Object-oriented methods produce systems that can be maintained and enhanced more readily.

Reusable Components: Object-oriented features such as inheritance and polymorphism lead to smoother, more efficient reuse of code.

Reality-Based Systems: The system is delivered closer to the users actual needs.

Data Accessibility: Object-oriented techniques allow for greater retrieval of data when needed. This is attributed to a thorough understanding of the users data and the relationship among the data items.

User Involvement and Ownership: Object-oriented techniques allow for greater involvement of the users in the system project. Therefore, users develop a sense of ownership in the system.

3.16 The Constraints Of Object-Oriented Techniques

Graham (1993) identifies the fact that reusability may not always be delivered due to tight project deadlines. Brown (2002) identifies the cost of migrating to object-oriented methods as a pitfall. The two major costs of migrating to object-oriented methods include the cost of *The Installed Base* and *Retraining*. The design of object-oriented systems will need to accommodate the needs of legacy systems that are in place. Staff retraining may also be required as programming staff may not be familiar with object-oriented methods.

3.17 Unified Modelling Language

UML (Unified Modelling language) is a third generation object-oriented modelling language that was developed to fuse the main object-oriented methodologies. UML is accepted as a standard methodology in the object oriented society.
The Unified Modelling Language (UML) is a unification of a number of earlier object-oriented modelling languages.

The three principle designers of Unified Modelling Language are Grady Booch, James Rumbaugh and Ivar Jacobson. They collaborated to combine the best features in their individual object-oriented analysis and design method into a unified method.

Priestley (2000) states that Unified Modelling Language is a language for expressing object-oriented design models, it is not a complete methodology, however its authors consider it to be suitable for use with iterative and incremental development process.

In the Unified Modelling Language, a system is represented using five different "views" that describes the system from distinctly different perspectives. Each view is defined by a set of diagrams. The following views are present in UML; (1) *Use Case View, (2) Design View, (3) Implementation View, (4) Process View and (5) Deployment View.*

Figure 3.10: The 4+1 View Model

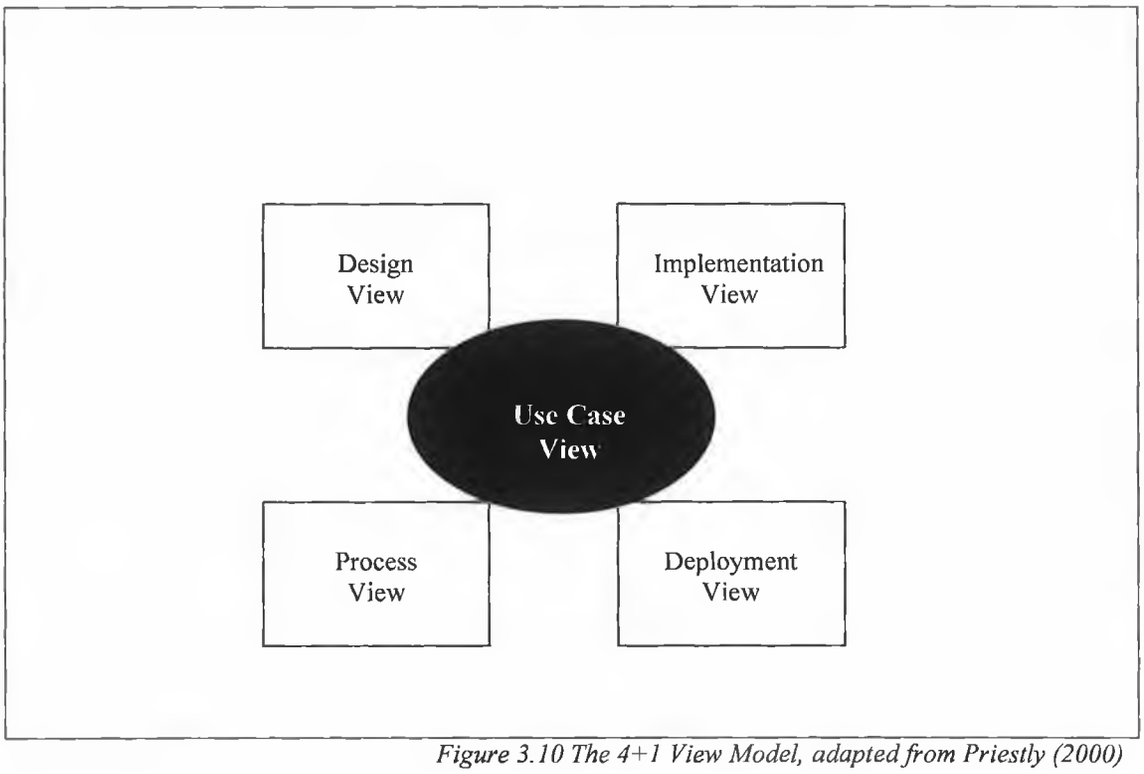


Figure 3.10 The 4+1 View Model, adapted from Priestly (2000)

Each view corresponds to a particular perspective from which the system can be examined. Different views highlight different aspects of the system that are of interest to particular groups of stake holders. Merging the information found in all five views can result in a complete model of the system.

3.17.1 The Use Case View

The use case view defines the systems external behaviour and is of interest to end users, analysts, and testers.

This view defines the requirements of the system, and consequently constrains all the other views, which describe certain aspects of the systems design or construction. This is why the use case view has a central role and is often said to drive the development process.

3.17.2 The Design View

This describes the logical structures that support the functional requirements expressed in the use case view. It consists of definitions of program components, principally classes, together with specifications of the data they hold, and their behaviour and interactions. Details of how the systems functionality will be implemented are described in this view.

3.17.3 The Implementation View

This describes the physical components out of which the system is to be constructed. These are distinct from the logical components described in the design view, and include such things as executable files, libraries of code and databases. The information contained in this view is relevant to activities such as configuration management and system integration.

3.17.4 The Process View And Deployment View

The process view deals with issues of concurrency within the system, and the deployment view describes how physical components are distributed across the physical environment, such as network the of computers that the system runs in. These two views address non-functional requirements of the system such as fault-tolerance and performance issues. The process and deployment views are relatively underdeveloped in UML, compared in particular with the design view which contains much of the notation that would informally be thought of as design-related.

UML defines nine distinct diagrams. These are listed in Table 3.9, together with an indication of the views that each is characteristically associated with. UML diagrams are largely graphical in form, because most people find such representations easier to work with rather than purely textual representations.

Table 3.9: UML's Diagram Types

	DIAGRAM	VIEW
1	Use Case Diagram	Use Case View
2	Object Diagram	Use Case And Design Views
3	Sequence Diagram	Use Case And Design Views
4	Collaboration Diagram	Use Case And Design Views
5	Class Diagram	Design View
6	Statechart Diagram	Design View
7	Activity Diagram	Design View
8	Component Diagram	Implementation View
9	Deployment Diagram	Deployment View

Table 3.9 UML's Diagram Types, adapted from Priestly (2000)

3.17.5 Use Case

For the purpose of this study, use cases will be discussed. Ivar Jacobson is regarded as the inventor of use cases. In his use case approach the use case model created in the analysis process serves as a basic for the models of the system developed subsequently. Chen (2001) states that the main reason that the concept of use cases has became so popular is that it bridges the gap between object oriented analysis and object oriented design. Some object-oriented methods have added the concept of use cases, while recently Jacobson's use case approach has been included in the Unified Modelling Language to capture the higher level of user functional requirements of the system.

Malan et al., (2001) states the following with regard to use cases:

> *"a use case defines a goal-oriented set of interactions between external actors and the system under consideration".*

Pressman (2000) states that a use case is a written narrative that describes the role of an actor as interaction with the system occurs. Use cases describe scenarios that will be perceived differently by different actors.

Priestly (2000) states that actors are as follows:

> *"The different roles that people fill when they interact with a system".*

Actors can model anything that needs information exchange with the system. Actors can model human users, as well as other systems that are going to communicate with the system (Jacobson et al., 1993).

The use case model is based on actors and use cases. It specifies the complete functional behaviour of the system by defining what exists outside the system (actors) and what should be performed by the system (use cases).

Each use case contains a complete sequence of related transactions performed by some actors and the system. The collections of use cases are the complete functional requirements of the proposed system.

Use cases are related by two associations; uses and extends. The extends association specifies how one use case description inserts itself into another use case description which is independent and unknowing of the former one. Extends can be viewed as a kind of "inheritance" between use cases. The uses association can be viewed as a kind of aggregation.

It is essential that both the users and developers, or the analysts and the domain experts, can communicate with each other and users can be involved throughout the process of system development. Use cases are an effective way of capturing both system requirements and business process.

The concept of use cases has become more and more popular in the object-oriented community and has been established as one of the fundamental techniques of the object-oriented methodology. It provides some advantages in the process of requirements elicitation, which the conventional object-oriented methods failed to achieve. Like many other development methods, there are still some problems when applying the use case technique in developing the system.

3.18 Summary

There is no single methodology that covers all aspects of systems development, however, some methodologies are more comprehensive than others. The systems development life cycle (SDLC) was developed to produce information systems on schedule, within a required budget, and to the requirements of the user. Information systems development can be divided into three perspectives; The Logical Process Perspective, the Data Perspective, and the Event Perspective.

Information Engineering methodology is deeply rooted in data analysis. Structured Systems Analysis and Design Method (SSADM) is based on the assumption that systems have an underlying data structure, which can change very little over time. Structured Systems Analysis and Design (SSADM) comprises of a hierarchy of modules, stages, steps and tasks, which serve to divide the development process into a number of phases.

Object-orientation is a technique for system modelling. Object-orientation makes it possible to model a system as a number of concepts that interact with one another. The Object-Oriented Development Life Cycle (OODLC) is the traditional development life cycle in a modern form. Object oriented techniques offer benefits such as system stability, maintainability, reusable components, accessibility, and user involvement and ownership.

However, there are many problems associated with object-oriented techniques. Reusability may not always be delivered due to tight project deadlines. There are two major costs when migrating to object-oriented method, which are the costs of of installation and training.

3.19 Conclusions

The emergence of the World Wide Web has given rise to a new generation of information systems. The World Wide Web provides a client-server architecture. The evolution of technology has given rise to applications that are constantly modified, enhanced navigation and increased interface features.

Object-oriented techniques are a key approach in building interactive applications, especially in the case of large scale projects. Web-based application development methodologies are based on traditional approaches. However, traditional approaches do not offer all that is necessary when developing Web-based applications.

The next chapter will review the most prominent Web-based application development methodologies, which encompass traditional techniques.

CHAPTER 4: WEB APPLICATIONS

4.1 Rapid Evolution Of The Web

The Web has changed from a novelty factor to more of a lifestyle accessory. It has matured to a point where it is providing a viable and expedient alternative to many of the traditional method of conducting business. Walsh (2000) suggests that the Web has evolved, moving from basic Web sites to complex eCommerce Web sites.
The Web has changed in the following ways:

- Basic Web sites lacking any special effects.
- ECommerce transactions for digital items such as software downloads.
- Complex eCommerce sites, selling physical goods such as products, e.g. www.bb-europe.com.
- Sophisticated sites that are tailored to customer's personal details and demographics.

Ginige et al., (2001a) discuss the importance of the World Wide Web:

> *"Within a short period, the Internet and World Wide Web have become ubiquitous, surpassing all other technological developments in our history."*

The use of the World Wide Web as a business tool is increasing at a rapid pace. The Web represents a paradigm shift in information technology and offers unprecedented opportunities to companies. The Web has characteristics that permit it to be an ideal platform for building business systems. The Web has *An Ubiquitous Global Network, Standards-Based Infrastructure*, and A *Universal Client* (White Paper, Allaire Corporation, 2000).

4.2 Move Towards Web-Based Applications

Anderson (2001) depicts the move towards Web-based applications in the following statement:

> *"Slowly I turn, step by step, inch by inch."*

This is a perfect analogy that depicts the slow move towards Web-based applications. For many years application vendors were unable to write Web-based applications due to the following;

- Development environments were non-existent.
- Few programming-languages and minimal scripting support was available.
- Developers were difficult to locate and retain.
- Browsers were immature.

However, the barriers to Web-based applications were removed and the move towards the latter began. This was attributed to technological advances within the industry.

Anderson (2001) states the following:

> *"The shift towards Web-based models has not happened overnight and will not be finished by tomorrow, but a significant headway has been made."*

There is a growing concern with regard to the manner in which Web-based systems are developed. Their development has been ad hoc, lacking a systematic approach, and quality and control assurance. There is very little attention given to the development methodologies, evaluation techniques and project management (Murugesan et al., 1999).

4.3 Introduction To Web Applications

Chen et al., (2001) provide the following generic definition of a Web application:

> *"A Web application is defined as any application/software program that runs on the Internet, or corporate intranets and extranets."*

A Web application is software that is used over the World Wide Web. It is an interactive application that can be accessed through a corporate intranet or through the Internet. A Web application can be viewed as an information system that uses World Wide Web technology to provide easy access to a wide variety of information resources for internal and external users.

Phanouriou (1996) defines Web applications as:

> *"Web applications are applications designed to allow any authorised user, with a WWW browser and an Internet connection, to interact with them. The application code usually sits on a remote server and the user interface is presented at the clients WWW browser. Web applications are by their nature platform independent and leverage off the accessibility of the Web."*

A Web application can be viewed as a system that performs specific functions directly for the user, or in some cases, for another application program. Frankel (1999) very simply defines Web applications as:

> *"Software programs that perform business functions such as accounting, word processing, spreadsheet, slide presentation, and database management."*

Pettit (2001) very simply defines a Web application as:

> *"The business logic that enables user interaction with the Web sites, and the transacting and interfacing with all the back-end data systems."*

Conallen (1999) attempts to establish that a Web application is a software system with business state:

> *"A Web application is a Web system (Web server, network, Http, browser) in which user input (navigation and data input) effects the state of the business."*

There are three forms of Web applications; *Static Web Documents,* which is a personal Web page, including pictures. *Interactive Web Applications,* a survey form on a Web site. *Complex Web database systems,* that can handle complicated business transactions online, such as electronic commerce and stock trading.

Web documents are grouped with regard to information change in a document. The information in a static document remains the same until the author reviews the document. The information in a dynamic document changes whenever a server receives a request of the document. Information displayed by an active document can change after the document has been loaded into a browser (Comer, 1997).

4.3.1 The Purpose Of A Web Application

The purpose of a Web application is the most crucial step in determining the achievement of the application, as with any information system. The most common use of Web applications include the following; Information Retrieval, Information Dissemination, Information Processing, and Technology Explorations (Artz, 1996).

Information retrieval applications facilitate access to reports, documents and procedure methods. Information dissemination applications furnish information to a wider audience, such as job listings or products. Information processing applications use Web technology to amplify the capabilities of structured information systems over the Internet. Technology exploration applications investigate the potential of the technology.

4.3.2 Objectives Of A Web Application

Burdman (1999) states that clear objectives are a very important factor in the success of a Web project. They are specific and attainable. Objectives are essential in the development of complex information systems. They assist in the organisation of the application and provide a problem statement. Artz (1996) identifies the following as typical categories of objectives; *Business Objectives, Information Objectives, Functional Objectives, User Interface Objectives, Development Objectives, Operational Objectives,* and *Quality Objectives.*

The business objectives analyse the function that the application should address in terms of the business perspective. The information objectives address the categories of information resources that the application should provide. Functional objectives examine the potential capabilities of the application. User interface objectives address factors that concern the user. Development objectives are the force behind the development process. Operational objectives identify the operating characteristics of the target system and take account of issues such as security, performance and audit ability. Quality objectives characterise the quality goals of the system and embrace reliability, maintainability, and flexibility.

4.4 Characteristics Of Web Applications

Web applications are composed of static HTML documents and programs that run at both server and client side. Hyper-Text Transfer Protocol (HTTP) is used for communication between Web browsers and Web servers (Yang et al., 1999).

Web applications can fluctuate from small-scale, to large-scale enterprise applications and they have evolved and become more elaborate. The complexity may be either in terms of performance or in terms of the dynamic nature of the information. Table 4.1 identifies the characteristics of simple and advanced Web-based systems.

Table 4.1: Characteristics Of Simple And Advanced Web-Based Systems	
Simple Web-Based Systems	**Advanced Web-Based Systems**
Simple Web pages primarily presenting textual information.	Complex Web pages.
Information content does not change – fairly static.	Information is dynamic – changes with time and users needs.
Simple navigation.	Difficult to navigate and find information.
Stand –alone systems.	Integrated with database and other planning, scheduling, and tracking systems.
High performance is not a major requirement.	Requires high performance and continuous availability.
Developed by a single individual or by a small team.	Requires a large development team with expertise in diverse areas.
Used for information dissemination in non core applications.	Deployed in mission-critical applications.

Table 4.1 Adapted from Ginige et al., (2001)

4.5 Evolution And Taxonomy Of Web applications

The order of the categories illustrates the evolution of Web applications. It is possible to initiate Web development with an application from any category. See Table 4.2

Table 4.2: Categories Of Web Applications	
Category	**Examples**
Informational	Online newspapers, product catalogues, newsletters, service manuals, online classifieds, online electronic books.
Interactive • User-provided information • Customised access	Registration forms, customised information presentation, online games.
Transactional	Electronic shopping, ordering goods and services, online banking.
Workflow	Online planning and scheduling systems, inventory management, status monitoring.
Collaborative work environments	Distributed authoring systems, collaborative design tools.
Online communities, marketplace	Chat groups, recommender systems that recommend products or services, online marketplace, online auctions.
Web portals	Electronic shopping malls, online intermediaries

Table 4.2 Adapted from Ginige et al. (2001).

4.6 The Components Of A Web Application

Hassen et al., (2001) discuss the following components that exist in Web applications. They include *Web browsers* (used by the clients), *Web servers*; *Web pages*; *Application servers*; *Application pages*; *Databases*; *Distributed objects*; and *Multimedia Web objects* such as images and video. See figure 4.1.

The user of the Web application uses the Web browser as the interface, which enables the user to gain access to the functionality of the Web application. The browser transmits the users actions to the Web server. Requests are sent via the HTTP protocol. When a request is received the Web server determines whether or not it can fulfil the request directly. If not, the application server must be involved. The application server processes the application page and returns an HTML page to the Web server. The Web server returns the requested page to the Web browser, which displays it to the user.

Figure 4.1: Data Flow Between The Components Of A Web Application

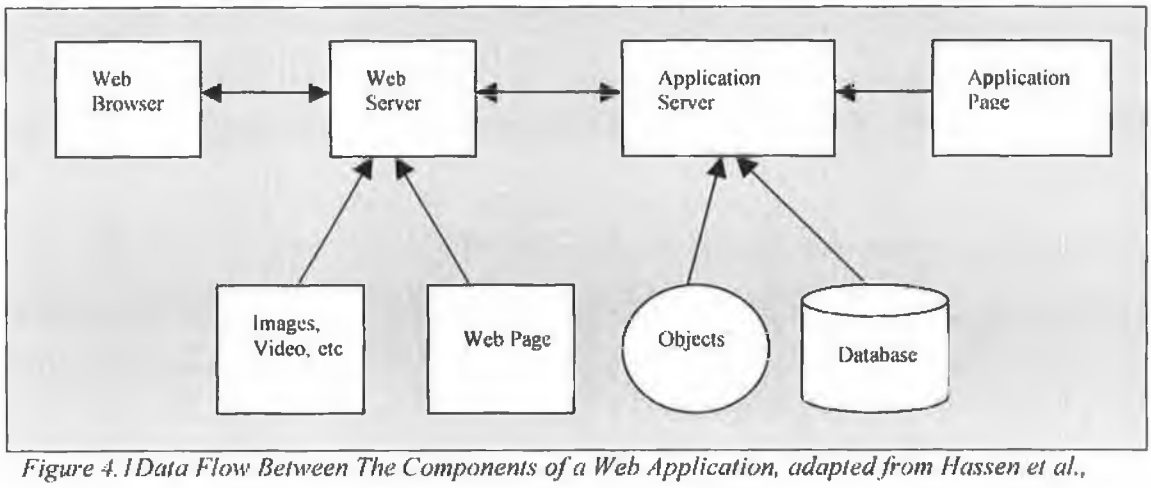


Figure 4.1Data Flow Between The Components of a Web Application, adapted from Hassen et al., (2001)

Web applications require a Web application server in addition to the HTTP server used for static sites. They provide both the development tools and runtime platform for developing and delivering applications on the Web. Application servers offer a range of services, including security and state management, as well as database connectivity and other service technologies. A Web server runs specialised Web server software that supports HTTP to handle multiple Web requests and is responsible for user authentication in intranet and extranet applications. The application server performs most of the application processing logic and enforces business rules. The database server hosts the database management system and provides data access and management capabilities. The database server performs the database query and sends the result back to the Web server.

4.7 Architecture Of Web Applications

Yang et al., (1999) state that a Web application uses a three-tier application architecture. Web Application architecture consists of three major tiers; *Web Browser*, *Web Server*, and *Optional Database Servers*. The information process in the application is passed through each tier. The user interaction is performed at the Web browser tier, the program logic at the Web server tier, and the database processing at the database server tier.

Pettit (2001) identifies that a Web application is more than the backend system that resides on an operating system. Pettit (2001) identifies the main components of a Web application as the following; User Interface Code, Web Server Software, Front End Systems, Backend Systems and a Database.

4.7.1 User Interface Code

The user interface code is the presentation layer of the Web application. This code formulates the appearance and feeling of the site. This code interfaces directly with the server software, and in addition provides client-side code that can generate semi-automated features and responses on behalf of the user directly to the server.

4.7.2 Web Server Software

The Web server supports the physical communication between the user browser and the application that the user needs to access.

It manipulates every part of inbound and outbound HTTP'S requests. It supervises the user sessions and endeavours to make sure all sessions are appropriately processed.

4.7.3 Frontend Systems

The frontend system interfaces directly with the user interface code, the operating system, and the backend systems. Under normal circumstances a user will not interface directly with this layer. However, the data the user passes to the user interface code will be passed through the frontend system.

4.7.4 Backend Systems

The backend systems are the real driving components of any Web application. The business needs drive the development of the backend systems, and the resulting code provides the business function, such as facilitating online transactions. User input is passed to this level via the user interface code and any associated frontend system. The backend systems interface directly with the frontend system, the operating system, the database, and possibly the data itself.

4.7.5 Database

A database is a collection of data that is organised so that its contents can easily be accessed, managed, and updated. The database controls the data that the Web application use and manage. An industrial strength Web database application may consist of five major components, as shown in figure 4.2.

Figure 4.2: Web Database Application Components

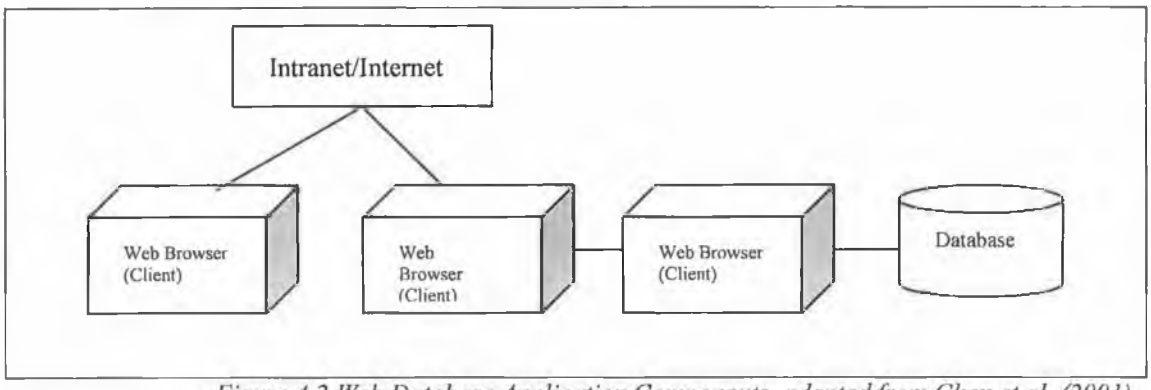


Figure 4.2 Web Database Application Components, adapted from Chen et al, (2001)

4.8 Types Of Web Applications

Frankel (1999) identifies the following types of Web applications; *Enterprise Resource Planning, Human Resource, Payroll and Accounting, Travel and Expenses, Graphic Applications, Office Suites, Productivity Applications* and *Virtual Desktops.* However, for the purpose of this study the following types of Web applications will be discussed.

4.8.1 Enterprise Resource Planning

Enterprise resource planning focus on supporting the supply chain process involved in the operations of a business, instead of focusing on the information processing requirements of business functions (O'Brien, 1999).

Frankel (1999) defines an Enterprise Resource Planning system as the following:

> *"A set of software modules that automate and streamlines functions such as inventory management, manufacturing, supply chain, human resources, purchasing and finances."*

4.8.2 Graphic Applications

Graphics are an integral part of a Web-based application. Graphics such as photos and illustrations create large files which are demanding on computer memory (Frankel, 1999). If the user needs are not overly elaborate, it is possible to take advantage of Web-based graphics programs without a significant investment of money (Simone, 1999a).

4.8.3 Human Resource Applications

(O'Brien, 1999) states that human resource information systems are designed to do the following:

- Support planning to meet the personnel needs of the business.
- Development of employees to their full potential.
- Control of all personnel policies and programs.

The Internet has become a major force for change in human resource management. Human resource management systems may recruit through corporate Web sites.

The two main providers of human resources via Web applications are PeopleSoft and Employease.com. PeopleSoft offer access to Human Resource applications via the Web. Employease.com offer employees self-service benefit plans (Roberts-Witt, 1999a).

4.8.4 Payroll And Accounting

Payroll and accounting systems produce data for tax purposes, such as reports to tax offices. If using a Web-based payroll and accounting application, the payroll processing is automated. Web-based accounting programs target small to medium sized businesses (Roberts-Witt, 1999b).

4.8.5 Virtual Desktops

Virtual desktops aim to give users a Web-based, server-stored version of applications such as e-mail, calendar, day planner, and file storage so that the user can access information anywhere at any particular time (Simone, 1999b).

Virtual desktops allow users to add new entries to an address list, update a schedule or even download a presentation. Frankel (1999) identifies some of the most common virtual desktops. These are as follows: Visto, StoragePoint, MagicalDesk, Zcentral, Desktop.com.

4.9 Advantages And Disadvantages Of Web Applications

Following is a table including the advantages and disadvantages of Web applications.

Table 4.3: Advantages And Disadvantages Of Web Applications

	ADVANTAGES	DISADVANTAGES
The Role of IT Changes:	Since applications are automatically updated, IT departments do not have the worry about supporting multiple versions of a product. The automatic upgrades will also deliver new drivers, new help files and new features.	On the downside, a company is essentially forced to upgrade, whether it wants to or not. Because traditional applications are stored locally, a company is free to use those applications for as long as it wants. Web applications are more like a service
Securing Data:	Many of today's Web-based applications allow data to be stored on local machines. This will reduce overhead costs.	This will reduce overhead costs, but it may leave your data vulnerable.
Save Money Now, Pay Later	Up-front costs will likely be lower for Web application but the long-term costs should even things out, or even prove higher. The Web application-pricing model usually involves putting some money down, then furnishing monthly payments to continue using the product.	A company that rarely upgrades its software, however, will likely end up paying more for Web applications.
High-Speed Access a Must:	Before a company decides to choose to go with any Web applications, they must ensure that they have high-speed Internet access in the first place. They must also consider whether the Web applications selected work over slower dial-up connections for telecommuters and road warriors.	Ensure that the ASP connection can deliver reliable access to applications.

Table 4.3 Adapted from Frankel (1999)

4.9.1 Advantages Over Traditional Non Web-Based Applications

(Chen et al., 2001) provide the following list of advantages of Web-based application over traditional non Web-based applications:

4.9.1.1 Control Over The Application

The Web application developer can govern the application on the server side for all users. The developer can readily control the code base and access to any part of the application. The application can become positively dynamic, from the binary execution to the available help. Developers can provide immediate updates and customisation.

4.9.1.2 Cross-Platform Capability

A HTML solution provides the ability to run an application on any Web browser on any operating system. Having cross-platform capability alleviates the developer from fretting about the client's configuration. If a client has a browser that can run Java code, it may not be necessary to identify what operating system the users have.

4.9.1.3 Control Over Versioning

A developer can control the right version of a file at the server, whether it is DLL, EXE or a database file. There is no longer a need to get the latest version of any part of the application out to the user; the client will always have the right code at the right time.

4.10 Web Application Development

Hypermedia systems have distinctive characteristics that separate them from traditional, more conventional systems.

These differences make a direct transposition of traditional techniques both difficult and inadequate (Nanard et al., 1995). The leading known process models such as the waterfall model, explorative process models, the prototype model and the spiral model pose serious constraints applicable to the development of Web applications. The powerful dynamics of change that apply to extensive and long-living Web applications, the distributed nature of the Web combined with the development process, lead to problems that impede the evolution of Web-applications.

The following section discusses methodologies that are in place to assist in the development of Web-based applications.

4.10.1 Specialised Techniques And Methodologies For Hypermedia Design

Following is a list of specialised techniques and methodologies for hypermedia design.

- The Hypermedia Design Model **(HDM)** technique (Garzotto et al., 1993). This method builds upon and integrates features of Entity-Relationship Modelling (Chen, 1976) and the Dexter Hypertext Reference Model (Halasz et al., 1990).

- **HDM-Lite** (Fraternali et al., 1998). This is an adaptation of the Hypermedia Design Model (HDM), specifically tailored to Web-based systems.
- The Relationship Management Methodology **(RMM)** (Isakowitz et al., 1995). The Relationship Management Methodology prescribes a formalised methodology, consisting of seven steps from design through to implementing and testing.
- The View-Based Hypermedia Design Methodology **(VHDM)** (Lee et al, 1999a). This is derived from Relationship Management Methodology. The View-Based Hypermedia Design Methodology is bottom up; it beings with an analysis of users information and navigational requirements.
- The Object-Oriented Hypermedia Design Method **(OOHDM)** (Schwabe et al., 1995). This is based on the Object Modelling Technique. It extends HDM by providing methodological guidance through four activities: conceptual design, navigational design, abstract interface design, and implementation.
- **OOHEM-Web** (Schwabe et al., 1998). This maps the OODHM methodology to a rapid prototyping environment for Web-based hypermedia systems.
- The Object Modelling Technique **(OMT)** (Rumbaugh et al., 1991). This method is useful for modelling hypermedia systems.
- Enhanced Object-Relationship Model **(EORM).** This is an extended version of the Object Modelling Technique (OMT) for hypermedia systems design.

- The Unified Modelling Language **(UML).** This is a successor of the Object Modelling Technique (OMT). The Unified Modelling Language includes techniques for use-case modelling, class/object-relationship modelling, dynamic modelling of interactivity and real-time aspects and architectural specifications.
- Scenario-based Object-Oriented Hypermedia Design Methodology **(SOHDM)** (Lee et al., 1998b). This method analyses scenarios. This methodology has similarities with the Relationship Management Methodology (RMM), Object-Oriented Hypermedia Design Method (OOHDM) and the Enhanced Object-Relationship Model (EORM).

Figure 4.3: Specialised Techniques And Methodologies For Hypermedia Design

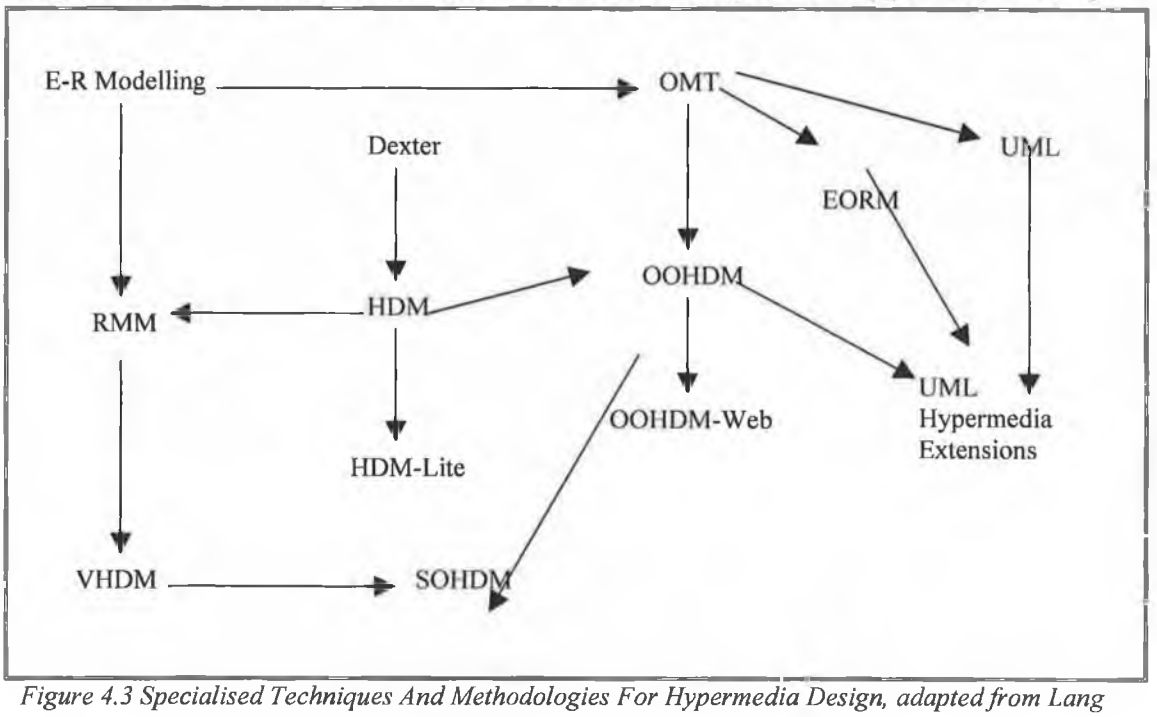


Figure 4.3 Specialised Techniques And Methodologies For Hypermedia Design, adapted from Lang (2001).

4.11 Web Site Design Method (WSDM)

De Troyer et al., (1998) define the Web Site Design Method as follows:

> *"WSDM is a user-centred method for the design of kiosk Web sites. By explicitly starting from the requirements of the users or visitors, WSDM solves Web site problems that are primarily caused by the fact that a site has no underlying design at all, or that the design is mostly data-driven."*

The Web Site Design Method is user-centred rather than data-centred design. The design is driven by the views of different user classes instead of the available data (Gaedke et al., 2000). Users are classified into user classes and the available data is modelled from the viewpoint of the different user classes. This results in Web sites that are better suited to the users and will therefore return greater usability and give extreme satisfaction.

The method consists of the following phases: *User Modelling, Conceptual Design, Implementation Design* and the actual *Implementation.* The User Modelling phase consists of two sub-phases: *User Classification* and *User Class Description.* The Conceptual Design phase also consists of two sub phases; *Object Modelling* and *Navigational Design* (Koch, 2001). See figure 4.4.

Figure 4.4: Overview Of The WSDM Phases

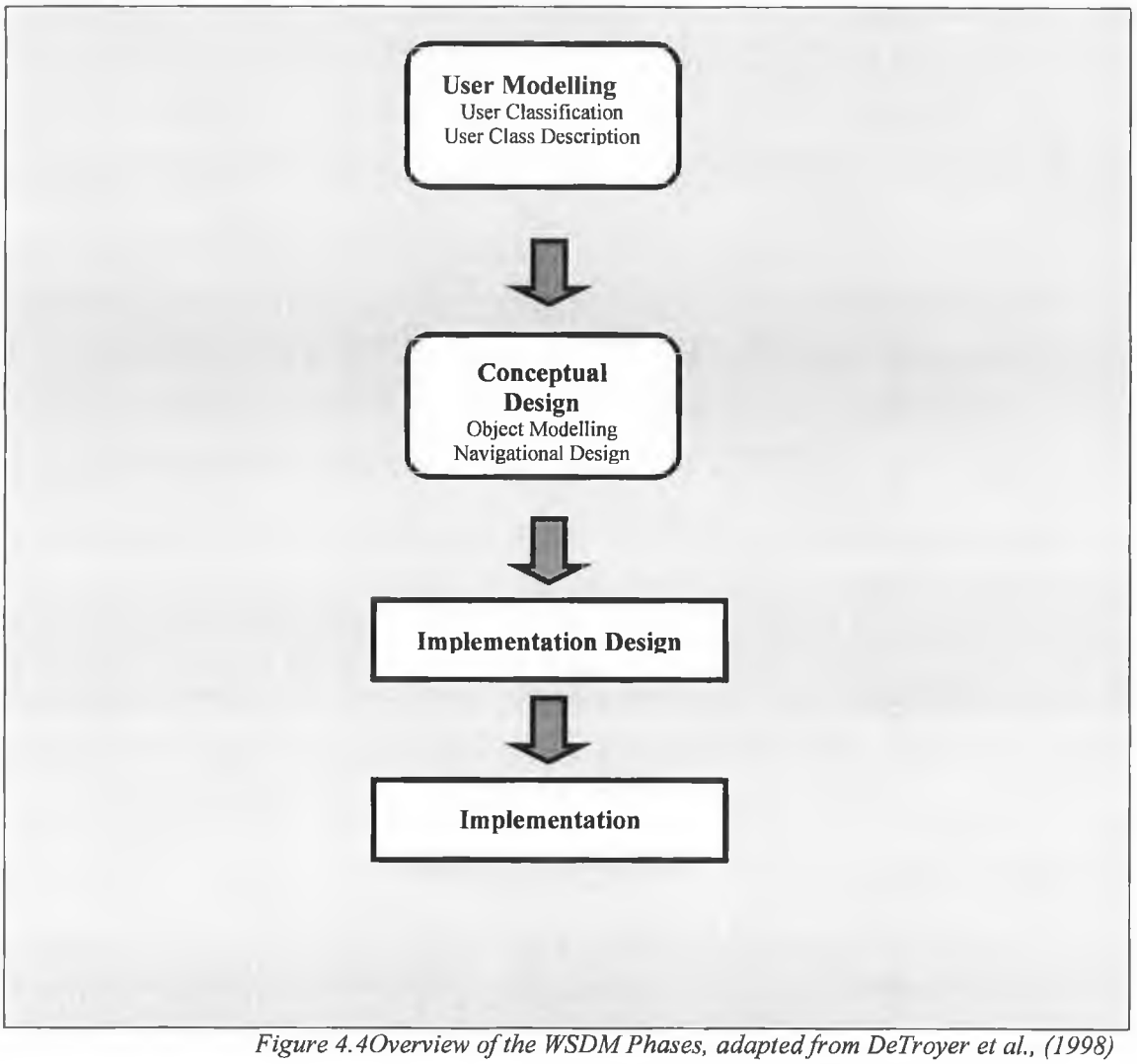


Figure 4.4Overview of the WSDM Phases, adapted from DeTroyer et al., (1998)

4.11.1.1 User Modelling

In the user modelling phase the potential users/visitors of the Web site are identified and classified according to their interests and navigation preferences. Different perspectives are defined for the user classes; these are different modes in which classes of users look and navigate through the information.

4.11.1.2 Conceptual Design

Conceptual design consists of two steps; *Object Modelling* and *Navigational Design.* During object modelling the information requirements of the different user classes and their perspectives are formally described. The navigational design phases describes how different users navigate through the Web site.

4.11.1.3 Implementation Design And Implementation

The *"look and feel"* of the Web site is created in this phase. The aim is to create a consistent, pleasing and efficient look and feel for the conceptual design made in the previous phase.

Implementation is the last phase of the Web Site Design Method (WSDM). It is the actual realisation of the Web site using the chosen implementation environment. This step can be largely automated using available tools and environments for assisting in HTML implementations.

4.12 Object-Oriented Hypermedia Design Method

The Object-Oriented Hypermedia Design Method is a model-based approach for building hypermedia applications. It is used to design different types of applications such as Web sites and information systems, interactive kiosks, and multimedia presentations (Schwabe et al., 1999b).

Schwabe et al., (1995) define the Object-Oriented Hypermedia Design Method as follows:

> "*The Object-Oriented Hypermedia Design Method (OOHDM) uses abstraction and composition mechanisms in the object oriented framework to, on one hand, allow a concise description of complex information items, and on the other hand, allow the specification of complex navigation patterns and interface transformations.*"

Object-Oriented Hypermedia Design Method (OOHDM) incorporates four various activities; *Conceptual Design*, *Navigational Design, Abstract Interface Design* and *Implementation.* They are accomplished in a mix of incremental, iterative and prototype based development styles. During each activity, except for the Implementation, a set of object-oriented models describing particular design concerns are built or enriched from previous iterations.

Considering conceptual, navigational and interface design as separate activities allows the designer to concentrate on different aspects of the development process at a particular time. Navigational design is the key activity in the implementation of Web applications and it must be distinctly separated from conceptual modelling.

4.12.1 Conceptual Design

The goal of the Conceptual Design activity is to construct a model of the application domain, using object-oriented modelling principles and primitives similar to those in Unified Modelling Language (UML). The product of this step is a class schema built out of Sub-Systems, Classes and Relationships (Schwabe et al., 1999b). Web application models are more than conceptual models.

Conceptual classes may be built using aggregation and generalisation/specialisation hierarchies. The main concern during this step is to capture the domain semantics as neutrally as possible, with slight concern for the types of users and tasks. If the application involves computations on objects, such as in Web-based information systems, the conceptual model will evolve into an object model that will be implemented in the target environment. In this case, navigation objects will act as observers over these application objects.

Object-Oriented Hypermedia Design Method (OOHDM) does not prescribe any particular method to produce the Conceptual Class Schema.

4.12.2 Navigational Design

In Object-Oriented Hypermedia Design Method (OOHDM), navigation is considered a critical step in the design of hypermedia applications. A navigational model is built as a view over a conceptual model, allowing the construction of different models according to different user profile.

Navigation design is expressed in two schemas; the Navigational Class Schema and the Navigational Context Schema. A schema containing navigational classes defines the navigational class structure of a Web application. In Object-Oriented Hypermedia Design Method (OOHDM) there is a set of predefined types of navigational classes; nodes, links, anchors and access structures. Access structures, such as indexes, represent possible ways for starting navigation.

4.12.3 Abstract Interface Design

The abstract interface design activity specifies which interface objects the user will perceive and how the interface will behave. There is a difference between navigation operations and interface operations; not everything that happens in the interface is navigation related. It is useful to design interfaces at an abstract level, to achieve independence of the implementation environment.

The abstract interface specification includes the way in which different navigational objects will look. It specifies which interface objects will activate navigation and the way in which multimedia interface objects will be synchronised and which interface transformation will take place.

User interface design is a critical activity in interactive applications, including hypermedia. Abstract data view is used in the design approach for describing the user interface of a hypermedia application.

4.11.4 Implementation

During the implementation activity the conceptual, navigation and interface objects are mapped onto the particular runtime environment that is being targeted. If the target implementation environment is not completely object-oriented, the conceptual navigational and abstract interface objects need to be mapped into concrete objects.

The designer will actually implement the design. Up to this point, all models were deliberately constructed in such a way as to be independent of the implementation platform; in this phase the particular runtime environment is taken into account. Table 4.4 below summarises the steps, products, mechanisms and design concerns in Object-Oriented Hypermedia Design Method (OOHDM).

Table 4.4: Steps, Products, Mechanisms And Design Concerns in OOHDM				
Activities	**Products**	**Formalisms**	**Mechanisms**	**Design Concerns**
Requirements Gathering	Use Cases, Annotations	Scenarios; User Interaction Diagrams; Design Patterns	Scenario and Use Case Analysis, UID mapping to Conceptual Model	Capture the stakeholder requirements for the application.
Conceptual Design	Classes, Sub systems, relationships, attribute perspective.	Object-oriented Modelling constructs; Design Patterns.	Classification, aggregation, generalisation and specialisation.	Model the semantics of the application domain.
Navigational Design	Nodes, links, access structures, navigational context, navigational transformations	Object-Oriented Views; Object Oriented State charts; Context Classes; Design Patterns; User Centred Scenarios	Classification, Aggregation, generalisation and specialisation.	Takes into account user profile and task. Emphasis on cognitive aspects. Build the navigational structure of the application.
Abstract Interface Design	Abstract interface objects, responses to external events, interface transformations	Abstract Data Views; Configuration Diagrams; ADV-Charts; Design Patterns	Mapping between navigation and perceptible objects	Model perceptible objects, implementing chosen metaphors. Describe interface for navigational objects. Define layout of interface objects.
Implementation	Running applications	Those supported by the target environment	Those provided by the target environment	Performance, completeness

Table 4.4 Adapted from Schwabe & Rossi, (1995).

4.13 Relationship Management Methodology

The Relationship Management Methodology facilitates the designer to use both top-down and bottom-up approaches. The development process is flexible and iterative and the quality of the final product is improved due to the fact that the development process is structured, extensible, maintainable and reusable (Costagliola et al., 2002). The Relationship Management Methodology uses a specific software tool, named RM-Case. This software supports the Relationship Management Methodology design and development and produces all associated diagrams required for the Relationship Management Methodology.

Relationship Management Methodology is focused on the design phase and employs the relationship management data model (RMDM), which represents the integration of Entity-Relationship and Hypermedia Design models. Relationship Management Design Model provides a language for describing information objects and navigation modality in hypermedia applications (Costagliola et al., 2002).

Diaz et al., (1995) defines the Relationship Management Methodology as follows:

> *"The Relationship Management Methodology is a methodology for the design and construction of hypermedia applications."*

Relationship Management Methodology concentrates on highly structured applications with great information volatility. It provides guidelines that can facilitate the automated design process of hypermedia applications. Relationship Management Methodology is based on the Entity-Relationship (E-R) model.

Gaedke et al., (2000) define the Relationship Management Methodology as follows:

> *"a platform-independent process-model for hypermedia applications that can also be used for the development of Web-applications."*

The Relationship Management Methodology (RMM) addresses the design and construction of hypermedia applications defining a process of seven steps. These steps are: *Entity-Relationship Design, Slice Design, Navigational Design, User Interface Design, Protocol Conversion Design, Run-Time Behaviour*, And *Construction And Testing* (Koch 2001).

4.13.1 Steps In The Relationship Management Methodology

Relationship Management Methodology describes completely the software development cycle, but focuses on the critical design phases. It does not address early and late lifecycle activities, such as project management, feasibility studies, requirement analysis, planning, evaluation and maintenance.

The methodology foresees an iterative approach to the development since it encourages feedback between the various development steps. Figure 4.5 highlights the phases of the Relationship Management Methodology.

Figure 4.5: The RMM Methodology

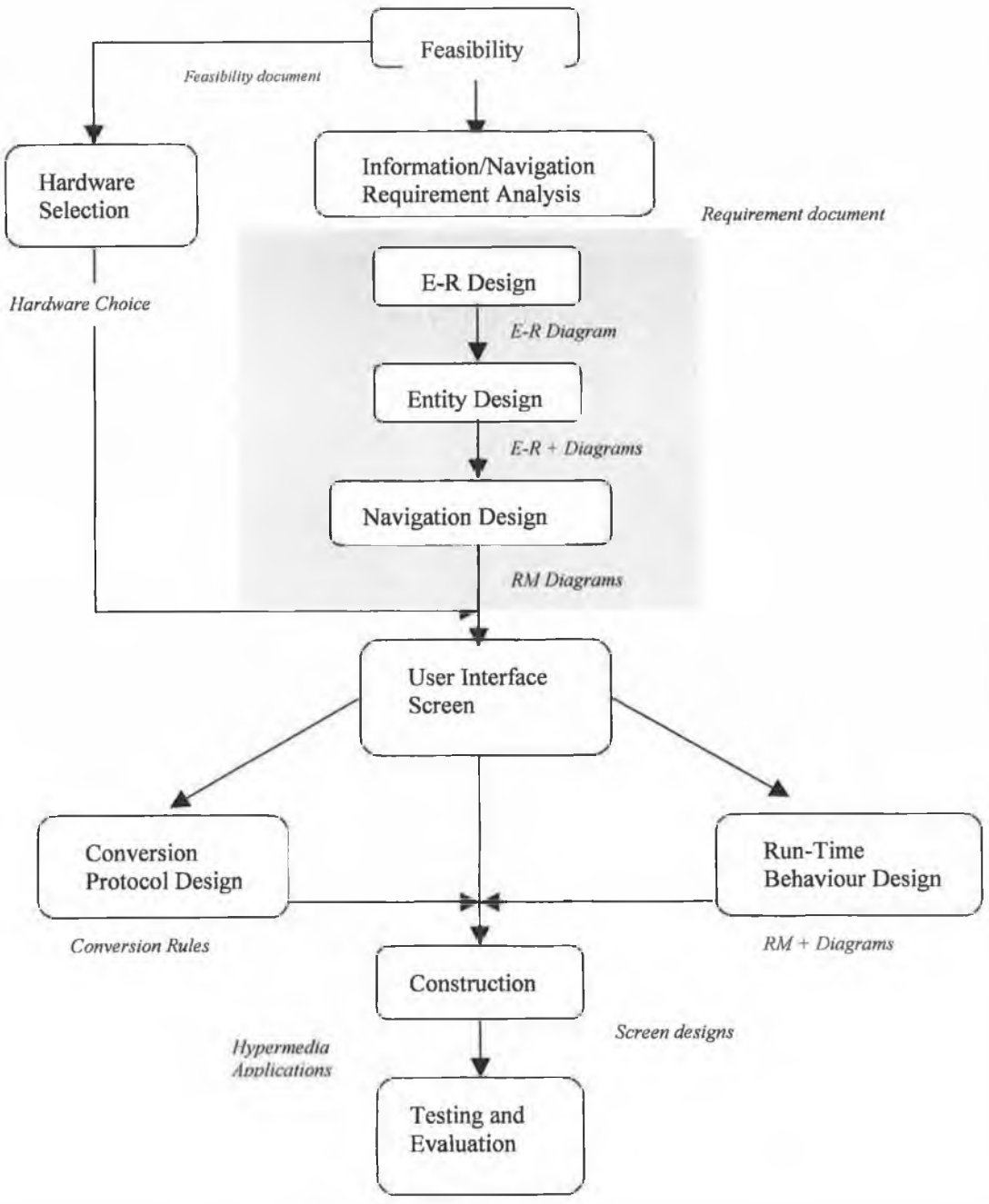


Figure 4.5 The RMM Methodology, adapted from Isakowitz et al., (1995)

4.13.1.1 Step 1: E-R Design

The first design step is used to represent the information domain of the application via an Entity-Relationship (E-R) diagram. This helps to determine a view of the application domain. This stage of the design process identifies entities, attributes and relationships of the application domain. These entities and relationships form the basis of the hypermedia application, and many of them will show up in the final application as nodes and links in a hypermedia Web.

4.13.1.2 Step 2: Slice Design

This step, unique to hypermedia applications, determines how the information is chosen. Entities are presented to the user. This stage involves splitting an entity into meaningful slices and organising them into a hypertext network. In its simplest form, all the information can be displayed within one window with scroll bars.

4.13.1.3 Step 3: Navigational Design

This phase allows the designer to establish how users will access information and the paths for the hypermedia navigation. Designers specify menu-like structures using indices and guided tours.

4.13.1.4 Step 4: User-Interface Design And Construction

User interface design involves the design of the screen layouts of every diagram element including access primitives, links, anchors, indexes and general navigational aids.

4.13.1.5 Step 5: Conversion Protocol Design

The conversion protocol design converts components of RMM into physical objects in the target hypermedia application. In this phase the designer describes how abstract constructs have to be transformed into physical-level constructs. For example, it is possible to define how a slice is converted into an HTML Web page.

4.13.1.6 Step 6: Run-Time Behaviour Design

This phase describes the functionality to be recognised at run-time. For example, possible inclusion of search engines, dynamic generation of pages, backtracking that allow the user to go back to previously visited nodes; history lists, maintaining an ordered list of each visited node (Costagliola et al., 2002).

4.13.1.7 Step 7: Construction And Testing

Extreme care must be taken when constructing and testing a hypermedia application. Special care must be taken in order to thoroughly test all of the navigational paths.

4.13.2 Comparison Of The Hypermedia Development Methods

Table 4.5 below identifies a comparison of the following; *Web Site Design Method* (WSDM), *Object Oriented Hypermedia Design Method* (OOHDM), and *Relationship Management Methodology (*RMM).

Table 4.5: Comparison Of Hypermedia Design Methods

	Process	Modelling Technique	Graphical Representation	Notation	Tool Support
RMM	1.E-R design 2.Slice design 3.Navigational design 4.Conversion protocol design 5.UI screen design 6.Run-time behaviour design 7.Construction and testing	E-R	1.E-R Diagram 2.Slice Diagram 3.RMDM Diagram	1. E-R 2. Own 3. Own	RMCase
OOHDM	1.Conceptual design 2.Navigational design 3.Abstract UI design 4.Implementation	OO	1.Class diagram 2.Navigational Class & Context schema 3.ADV Configuration diagram & ADV charts	1.OMT/ UML 2.Own 3.ADV's	OOHDM -Web
WSDM	1.User modelling 2.conceptual design 2.1 Object modelling 2.2 Navigational design 3.Implementation design 4.Implementation	E-R/OO	1.E-R or Class Diagram 2.Navigation layers	1.E-R/OMT 2.Own	

Table 4.5 Aadapted from Koch (2001)

4.14 Challenges Of Web Application Development

The design of hypermedia and Web systems present challenges and opportunities that are not normally encountered in conventional systems development (Rossi et al., 1999). Multi-tier Web-enabled applications can be complex, using a combination of technologies such as application servers, Web servers, traditional programming languages, scripting languages and multiple runtime environments. (White Paper, Compuware Corporation, 2002). Web application development presents challenges for Web application developers. Chen et al., (2001) discuss the following challenges of *Scalability, Load Balancing, Security,* and *Content Rich.*

4.14.1 Scalability

White Paper, Allaire Corporation (2000) state the following:

> *"Web application servers must be able to easily and affordably scale to handle any level of demand."*

Web applications must deliver high availability. Internet applications run on an open environment where the workload and user profiles are variable. Internet applications encounter highly variable and potentially enormous peak loads. The system needs to be designed to handle dramatic fluctuations of user demand as well as having additional upgrades to boost the system performance and to support additional users.

4.14.2 Load Balancing

Web application servers need to provide adaptable facilities for promptly and readily integrating with existing technologies and incorporating new technologies as they emerge (White Paper, Allaire Corporation, 2000).

In a multi-server Internet application, an unbalanced workload reduces system performance, reliability, and availability. Balancing the systems workload requires careful selection from an array of tools and techniques.

4.14.3 Security

Security is a major concern for Internet applications because of the open operating environment. Security must be designed into an application and must be maintained in that application. Security policies and procedures must be in place.
The following security issues need be addressed (Fournier, 1999):
Privacy, Integrity, Authentication, Access Control, Non-repudiation.

4.14.4 Content Rich

Content-rich applications require frequent updates and maintenance. A less frequently updated Web site quickly casts mistrust on its visitors mind with regard to its accuracy and usefulness. For Web applications, the notation of maintenance takes on a different meaning. The lines between development and maintenance blur to the point where they are really the same thing.

4.15 Web Engineering: An Introduction

Traditional methodologies do not work well for Web development as Web applications handle information in many forms, e.g. (text, graphics, video and audio), due to the fact that they are multidisciplinary. The traditional methodologies fall short in terms of the complexity and the scale of Web application projects. Callaway (1997) describes Web application development as a "*Mad Science*". The difficulties surrounding Web-based system development include the continuing advances in Web technology, the increase in commercial Web applications, and the migration of legacy systems to Web environments (Ginige et al., 2001a).

4.15.1 Web Engineering: The Needs And Principles

(Gaedke et al., 2000) state the following with regard to Web engineering:

> *"Web engineering promises to both reduce costs and increase quality during the development and evolution of Web applications."*

Web engineering supports a process and a systematic approach to the development of Web applications. Web engineering has the potential to bring the madness of Web application development under control, minimise risks and enhance maintainability and quality. Murugesan et al., (1999) provide the following definition of Web engineering:

"Web engineering is the establishment and use of sound scientific, engineering and management principles and disciplined and systematic approaches to the successful development, deployment and maintenance of high quality Web-based systems and application."

Gaedke et al., (2000) provide the following definition of Web engineering:

"The application of systematic, disciplined and quantifiable approaches to the cost-effective development and evolution of high quality applications on the World Wide Web."

4.15.2 Web Engineering And Software Engineering

Web engineering involves some programming and software development, and adopts some of the principles of software engineering.

Web-based system development is different from software development, and also Web engineering is different from software engineering. Following is a list of the differences between Web development and software development.

- Web-based systems are document-oriented containing static or dynamic Web pages.
- Web-based systems will continue to be focused on the look and feel, favouring visual creativity and incorporation of multimedia in presentation and interface.
- Web-based systems will continue to be content driven.
- Web-based systems need to cater for users with diverse skills and capability, complicating human-computer interface, user interface and information presentation.
- The nature and characteristics of the medium of Web is not well understood as the software medium.
- The Web exemplifies a greater bond between art and science than generally encountered in software development.
- Most Web-based system need to be developed within a short time, making it difficult to apply the same level of formal planning and testing as used in software development.

There are many differences between Web development and software development as discussed above, however the are also many similarities between them. These include; *Need for methodologies, Requirements Elicitation, Programming, Testing* and *Maintenance* (Deshpande et al., 2002)

4.15.2.1 Methodologies

A recent survey on Web-based project development by the Cutter Consortium (reported in its Research Briefs, 7 November 2000) underlines serious complications plaguing large Web-based projects;

- Delivered systems did not meet business needs 84% of the time.
- Scheduled delays plagued the projects 79% of the time.
- Projects exceeded the budget 63% of the time.
- Delivered systems did not have required functionality 53% of the time.
- Deliverables were of poor quality 52% of the time.

4.15.2.2 Requirements Elicitation

Inadequate requirements specifications and steady evolution are two major contrasts between Web applications and other software. User-centric approaches and methods to build applications have an unrealised potential in arriving at better specifications. The freedom of the Web makes it viable to obtain user feedback on-line as opposed to more traditional methods, such as meetings, interviews, paper-based surveys and focus groups.

4.15.2.3 Testing, Metrics And Quality

Web testing has numerous dimensions in addition to traditional software testing.

Each unit of a Web application such as page, code, site, navigation, standards, and legal requirements must be tested. Usability testing has also become an enormous and somewhat debatable issue.

Web metrics and quality are interlinked, and like software metrics, under-utilised. However, additional tools are becoming accessible and Web engineers need to develop sensible policies to test their sites and applications.

4.15.2.4 Maintenance

Web maintenance is a continuous activity, more so than software maintenance. The maintenance can become complex depending on the nature of the application and it does not solely reside in the technical domain. The major differences between Web application and software maintenance arise in relation to content management and site navigation, apart from the evolutionary aspect.

Content generation will vary across organisations and applications. Human resources may carry out the allocation of responsibilities for content. It is imperative that Web developer's detail how content maintenance should be carried out and more importantly influences the relevant policies (Deshpande et al., 2002)

4.16 Web Application Development Tools

Over the past few years Web design programs have evolved from undependable programs to sophisticated, stable applications that make Web design easy for every developer. Morris (1999) states, "*Like any craftsman, a Webmaster or Web designer needs a set of good quality tools.*"

The majority of development methodologies are affiliated with specific tools. Table 4.6 summarises four broad classes of tools commercially available for Web application design. What You See Is What You Get (WYSIWYG) commercial Web development environments, such as Macromedia Dreamweaver or NetObject's Fusion, concentrate overwhelmingly on the graphics design. Tag-based design environments such as Allaire's Homesite furnish support for rapid development of HTML and CFML code.

Table 4.6: HTML Development Tools		
TOOL	**PROS**	**CONS**
Hand Coding	• Provides Most Controls • Able To Use Latest Tags	• Slow Process • Prone To Making Errors • Requires Knowledge Of HTML • Difficult To Pre-Visualise A Page
Tag Editors	• Faster Than By Hand • Enable The Insertion Of Tags With Appropriate Syntax • Enable Editing Of Attributes And Links	• Requires Knowledge Of HTML • Difficult To Pre-Visualise A Page • Syntax Checkers Might Feel Constrictive
HTML WYSWYG	• Enables Visual Design Of Pages • Easy To Use • Do Not Require HTML Knowledge	• Provides The Least Amount Of Control Over The Code • Contains Odd Nuances • Difficult To Edit By Hand • Built-In Display Mode For Pages That Do Not Match Pages As Actually Rendered By Browsers
Translators	• Able To Convert Existing Documents Quickly	• May Still Require Hand Or Editor Cleanup

Table 4.6 Adapted from Rodriquez-Medina et al., (2001)

Fortunately, there are a number of applications which have been developed to assist in the development of Web pages.

These applications make it possible for anybody to create and publish on the Internet. The following is a list of possible tools; *Java Technologies*, *Simple Text Editors, HTML Enhanced Text Editors, WYSIWYG HTML Editors, Browser-Based HTML Editors* and *Web Site Managers.*

4.16.1 Java Technologies

The Java programming language was developed as an object-oriented programming language loosely based on C++ that would be processor-independent.

Fischman et al., (1996) define Java as follows:

> *"Java is a programming language that allows a user to retrieve little programs called "applets" from remote sites on the Web and run them on a local machine without worrying about software compatibility."*

Tribble (1996) defines Java as:

> *"an application development platform providing a portable, interpreted, high-performance, simple and object-oriented programming language and runtime environment."*

Java applications and applets start as Java language source code. This source is then compiled to byte code, which is stored on a server. In order to execute a Java application/applet, the user invokes a Java Virtual Machine (VM), which executes the Java language byte code. Java ensures that applications are available to all designated users from any location via customisable Web browser-based front ends (Tribble, 1996).

4.16.1.2 Problems Solved By The Java Platform

There is a great amount of dissatisfaction associated with enterprise computing. The constant administration and upgrading of desktops are expensive. Computing is not secure enough, incompatible desktops prevent universal access to applications and applications take too long to develop and deploy (Tribble, 1996). Java computing can address all of the above problems.

White Paper (1998) identifies the following problems addressed by Java Platform: *Interesting Web Pages; Security; Client-Server Computing; Platform-Independence Computing; Extensible Through Platform-Specific Additions.*

4.16.2 Simple Text Editors

At the most basic level, an HTML document is an ordinary text file. It can be formulated using simple text editors by anyone who has knowledge of the Hypertext Markup Language (HTML).

Examples include "Notepad" in MS-Windows. Simple text editors give complete control over the HTML document, as they are not limited to the ability of a particular Web development tool.

4.16.3 HTML Enhanced Text Editors

Text editors that have HTML extensions are designed to make it easier to develop Web pages. HTML Enhanced Text Editors assist by reducing common inaccuracies such as typographical errors. Examples include Notetab and HTML-Kit. It is important to note that HTML enhanced text editors are not 'What You See Is What You Get' (WYSIWYG) development tools.

4.16.4 WYSISYG HTML Editors

A WYSIWYG application is an application that enables the developer to see on the display screen exactly what will appear when the document is printed.
This differs from word processors that are incapable of displaying different fonts and graphics on the display screen even though the formatting codes have been inserted into the file. WYSIWYG editors make it easier for people to learn and use Web design applications. However, due to the fact the WYSIWYG editors are more complex to develop, they often do not support the latest trends in Web innovations.

4.16.5 Brower-Based HTML Editors

Some Web editors are designed to run as part of the Web browser. These are known as *"Brower-Based HTML Editors"*. This might be achieved as a Java applet downloaded to your computer, or as a program that lets the developer enter information into forms, which then writes the Web pages for them. This can be achieved by using various server side languages like Perl, ASP, PHP and Coldfusion.

4.16.6 Web Site Managers

A Web site manager is a tool that not only allows the developer to edit individual pages, but also allows the developer to manage the overall Web site. For example, a style sheet can be quickly changed that is used in a Web site. Often these tools include WYSIWYG capabilities, as well as other advanced editing and HTML verification features. Examples include, Macromedia Dreamweaver, Microsoft Frontpage 2002, Adobe GoLive.

The following is a list of tools for various levels of users: Macromedia Flash, Macromedia Dreamweaver, and Visual Basic .NET.

4.16.1.1 Macromedia Flash

(Perfetti, 2002) states, *"Flash is a powerful tool that offer developers huge capabilities."* Flash offers developers the power to create Web applications with more sophisticated client and server side interactivity. Flash is a serious contender in the Web application race.

4.16.1.2 Macromedia Dreamweaver

Benedetti (2002) states, *"There are literally dozens of Web authoring applications available, however, only a few equal the sophistication and ease of use found in Macromedia Dreamweaver MX".* Macromedia Dreamweaver includes all the tools necessary to develop Web pages, without the need for the developer to have any programming skills. Dreamweaver uses visual layout tools to make the process both intuitive and powerful. It supports the latest Web and server technologies. First-time users will find Jumpstart design and production tools very handy. Dreamweavers setup Wizard gives users the ability to quickly create dynamic, staged, or ISP-hosted sites.

4.16.1.5 Visual Basic .NET

Visual Basic .NET 2003 is a comprehensive tools for rapidly building Microsoft. NET connected applications for Microsoft Windows and the Web, dramatically increasing developer productivity, and enabling new business and enterprise opportunities. There are three editions of the Visual Studio.NET: Enterprise Architect Features, Enterprise Developer Features, and Professional Features.

Enterprise Architect provides comprehensive tools for architecting and building distributed applications, as well as designing, specifying, and communicating applications architecture and development best practices.

Enterprise Developer provides enterprise development teams with powerful tools for building mission-critical applications that target any device and integrate with any platform.

Professional enables the developer to rapidly build the broadest range of applications for Windows, the Web and mobile devices.

4.20 Summary

The Web presents a paradigm shift in information technology and offers unprecedented opportunities. The Web has characteristics that permit it to be an ideal platform for building business systems. There is a growing concern with regard to the manner in which Web-based systems are developed. Their development has been ad hoc, lacking a systematic approach. Web applications include Web browsers, Web servers, Web pages, Application servers, Application pages, Databases, Distributed objects and Multimedia Web objects.

There are many types of Web applications: Enterprise Resource Planning, Human Resources, Payroll and Accounting, Travel and Expenses, Graphic applications, Office suites, Productivity applications and Virtual Desktops.

Hypermedia systems have distinctive characteristics that separate them from traditional, more conventional systems. Many methodologies have been put forward to assist in the development of Web-based applications. They include the following: Web Site Design Method (WSDM), Object-Oriented Hypermedia Design Method (OOHDM), and Relationship Management Methodology (RMM). Web application development presents challenges for the developers such as Scalability, Load Balancing and Security.

The primary focus of Web engineering is to deliver a systematic approach to the development of high quality Internet and Web-based systems. It achieves this by dealing with all aspects of Web-based system development.

The following are applications that are used in the development of Web-based applications: Macromedia Fireworks, Macromedia Dreamweaver and Visual Basic .NET.

4.21 Conclusions

Developing Web-based systems is significantly different from traditional software development. A solid infrastructure must be in place to uphold the surge of Web-based systems in a controlled, but flexible manner. Web engineering provides such an infrastructure.

The importance of and the need for Web engineering is now reasonably established through a consensus among experts (Deshpande et al. 2002). The main themes of Web engineering encompass how to successfully manage the diversity and complexity of Web application development, and therefore how to avoid potential failures that may arise. The need for Web engineering is high.

The next chapter will review the research question and design methodology used throughout this research.

CHAPTER 5: THE RESEARCH QUESTION AND DESIGN METHODLOGY

5.1 Introduction

Chapters two and three have placed the study of information systems development in an historical viewpoint. Those who develop Web-based applications or Web-based information systems encounter significant problems. The quests for the potential to resolve Web application development problems have been described. Chapter four introduces the Web engineering paradigm, which is the most optimistic paradigm to date for the development of Web applications.

The main objective of this chapter is to examine the theoretical and conceptual considerations affecting the research design adopted by the author to complete this study. The research approach is then examined. Following this, the research design is identified and case studies are analysed. The fieldwork section analyses the procedure the researcher followed while conducting the case studies. The research is designed to analyse the approach taken by the subject developers regarding the development of Web-based applications.

5.2 Research Objective

The research is designed to analyse the methodologies, techniques and tools used in the development of Web-based applications. The research objectives are discerned as a primary objective, which answers the research question, and secondary objectives that qualify the answer.

5.2.1 The Primary Objective

The primary objective of this research is to provide:

- An evaluation of the approach taken by the subject developers with regard to the development of Web-based applications.

It is planned to achieve this objective by analysing the use of Web specific methodologies, techniques and tools used in the development of Web-based applications.

5.2.2 Secondary Objective

The secondary research objectives are presented in similar sequence to their related questions in the questionnaire. (See Table 5.1). These are as follows:

- To develop a profile of Web application development departments.

This objective will be achieved through reading academic papers and journals.

- To determine the current and planned usage of Web application development methodologies. (In house or otherwise).

This objective will be achieved by questioning interviewees regarding the confidence they have in the methodologies currently in use.

- To determine the current and planned usage of Web application development tools.

This objective will be achieved by questioning interviewees regarding the confidence they have in the development tools in use by them.

5.3 The Research Method

Having determined the overall goals of the study, the need to identify a research method was apparent. While considerable debate surrounds the choice of paradigms that might provide the best guidance for conducting research, two paradigms in particular have earned widespread use, the Positivist (Quantitative) and Phenomenology (Qualitative) approach.

Hoepfl (1997) describes qualitative methods as follows:

> *"Phenomenological inquiry, or qualitative research, uses a naturalistic approach that seeks to understand phenomena in context-specific settings."*

Qualitative research relies on transforming information from observations, reports and recordings into data in the form of written words. Techniques used by researchers using a qualitative approach include, Case Studies, In-depth interviewing, Participant observation and Diary Methods.

Hoepfl (1997) describes quantitative methods as follows:

> *"Logical positivism, or quantitative research, uses experimental methods and quantitative measures to test hypothetical generalisations."*

Quantitative research aims to measure phenomena so that they can be transformed into numbers. The main method used to collect quantitative data include, Interviews, Tests/Measures, Observation and Questionnaires.

It was decided to conduct case studies to elicit the required information relevant to this research. This approach is considered the most feasible because of the nature of the investigation. *"The decision to use a case study approach is a strategic decision that relates to the scale and scope of an investigation"* (Denscombe, 2003).

The case study approach encourages the use of multiple methods in order to capture the complex reality under scrutiny. Therefore, it was decided to conduct four interviews, based on a questionnaire. The questionnaire content assisted in providing detailed information on particular points of interest relevant to the research. Six companies were contacted with regard to participating in the research, and of those four agreed to be interviewed.

5.4 The Research Instrument

For the purpose of the interviews, a questionnaire was designed to identify information with regard to methodologies, techniques and tools used in the development of Web-based applications. The questionnaire was designed in consideration of, and in strict line with the research objectives. Each question was designed in light of the relevant literature.

5.4.1 The Questionnaire Content

The questionnaire content for organisations is summarised in Table 5.1.

Table 5.1: Questionnaire Content	
Question Number	**Question Purpose**
Q1 – Q4	Identify background information.
Q5	Identify the percentage of systems that were developed. Web-based applications or Web sites.
Q6	Identify increase, decrease of stability of systems developed.
Q7 – Q12	Identify complexities with regard to required functionality, data structures and user interfaces.
Q13	Identify the reasons for user interface design complexities.
Q14	How does each company deal with user interface design complexities.
Q15	Identify previous and current methodologies used.
Q16	Identify the perceived strengths of the methodology.
Q17	Identify the weaknesses of the methodology.
Q18	Identify previous and current development tools used.
Q19	Identify the perceived strengths of the tools.
Q20	Identify the weaknesses of the tools.
Q21	Identify previous and current development techniques used.
Q22	Identify any specific methodology planned for the future.
Q23	Identify reasons for adopting a new methodology.
Q24	Identify any development tool planned for the future.
Q 25	Identify reasons for adopting a new development tool.
Q 26	Identify any specific structured technique planned for the future.
Q 27	Identify reasons for adopting a new structured technique.
Q 28	Identify standards and guidelines.

5.4.2 Pretesting the Questionnaire

The questionnaire was presented to two project managers. They were asked to offer an evaluation of the questionnaire with particular emphasis on the following:

- The scope and content of the questionnaire, i.e. are all key issues identified, or are there any issues that would enlighten the research.
- The relevance of the questions, are they meaningful to practitioners? (Especially Web application developers).
- The wording of the questions.
- The order of questions.
- The length of the questionnaire.

Comments from the two evaluations resulted in only minor changes in the questionnaire content. One suggested that the order of the questions needed to be rotated slightly. This was taken into consideration and the relevant changes were made to the questionnaire.

5.5 Fieldwork

Having decided on the research design, the next step in the research process was the collection of data. Following the identification of a list of suitable candidates, the organisations were initially contacted by letter and subsequently by telephone. The potential respondents were assured of the importance of the research and the value of their support. An interview was agreed upon, and the respondents understood that the meeting was intended to produce material that was to be used for research purposes. All the respondents were offered a summary of the findings.

The respondents understood that their words could be treated as *'on the record'* and *'for the record'*. The venue for the interview was arranged, and the method for recording the interview was decided upon. All interviews were recorded by audio tape. Of the six organisations contacted, some felt that they did not have the time available to partake in such a study.

One organisation that had initially agreed to partake in the study and did participate in the study stage, later decided to be omitted from the study completely. The withdrawal came after substantial work was completed and at a time when it was too late to substitute the company.

The case studies consist of Web-based application developers who are specifically engaged in the development of Web applications. It was decided to conduct case studies on carefully selected Web application development organisations. The objective was to achieve the research objectives and to uncover other issues that may not have been addressed in the literature. It was also decided that should the researcher uncover any unexpected phenomena that a telephone survey would ensue. The purpose of the latter was to determine whether or not the results could be extrapolated to the wider development community.

5.6 Limitations Of The Research Design

This research is subject to all the limitations imposed by use of a case study, such as criticisms in relation to the credibility of generalisations, and in particular, lack of access to relevant documents.

The usefulness of the research is very dependent on the interviewees understanding of "Web-based applications". A serious limitation to the study, was the fact that one major organisation decided to withdraw from the research at a late stage and at a time when it was not feasible to introduce a replacement organisation to the study.

5.7 The Contribution Of The Research

The contribution of the case studies is envisaged as follows:

- An evaluation of the diversity of development methodologies, techniques and tools previously used, currently used and future use by the target companies.
- An evaluation of the methodologies used for the development of Web applications.
- A platform for future research in this area.

This study is expected to set a marker to which further research into Web specific development methodologies can be measured against.

CHAPTER 6: RESEARCH FINDINGS

6.1 Introduction

The research findings are based on the analysis of three case studies; NovaVision, Martec and Calico Commerce. A preliminary profile of each company is presented. The primary research question is addressed in terms of the past, current and planned usage of methodologies, techniques and tools in the development of Web-based applications. The case study questions are exploratory in nature, due to the deficiency of information in the field of Web-based application development.

This study therefore is not based on any pre-existing hypothesis. However, the rationale behind the study is based on the rapidly changing nature of the World Wide Web and the technologies and challenges that it presents. Finally, the research is concluded as several areas for further research are suggested.

6.2 Overview Of NovaVision, Martec & Calico Commerce

The following section provides an overview of the companies involved in the research.

6.2.1 NovaVision

NovaVision are a Web Design company based in Galway. They currently employ less than 50 staff. The company completed their first Web project in the late 1990's. They specialise in *Systems Integration*, *eBusiness* and *Web Development*. They design "*visually attractive*", innovative and dynamic Web sites.

NovaVision's core business is the design, implementation and delivery of complex enterprise systems utilising Internet technologies. They also provide interactive marketing consultancy. Examples of visually attractive Web sites designed by NovaVision include the following:

www.esatbt.com, www.bastaparsons.com, and www.baxolve.com

6.2.2 Martec

Martec are based in Galway, and specialise in providing Internet Web site design and development services. They employ less than 50 employees across a wide range of skills and services. Martec was established in the late 1990's. Their core business areas include *Internet Expertise*, *Information Security* and the development of *Customised Software*. Additional services provided by Martec include the following; Internet Consultancy, Corporate Branding, Visual Design, Information Architecture, Content Management, Internet Marketing.

6.2.3 Calico Commerce

Calico Commerce commenced trading in the mid 1990's. They are a large American company. They specialise in the development of *eBusiness Suites* that enable corporations to sell products and services through the Web. Their customers include leaders in high-technology, hardware, manufacturing, telecommunications services, financial services, home furnishings, chemicals, agriculture and retail.

Calico Commerce eBusiness suites include: *Loyalty Builder, Advisor, Configurator, Market Maker, Quote and Connector.*

For the purpose of this study, NovaVision, Martec and Calico Commerce may be categorised according to features of size, type and systems developed. The cases are analysed based on the methodology, techniques and tools within an organisational context. See Table 6.1

Table 6.1: Category Of Company

Company	**Category**
NovaVision	*Visually Attractive Web-based Application Development*
Martec	*Functional Service Based*
Calico Commerce	*E-Business Suites Development*

Table 6.1 Category of Company

The choice of NovaVision for this case study is based on their experience in developing "*visually attractive*" Web Sites and Web-based applications for an Irish and European market. They, therefore provide an opportunity to explore how a company with previous Web site design experience, approach the development of Web-based applications.

The choice of Martec for this case study is based on their knowledge of Web site development, combined with the wisdom they have as a functional service provider.

Martec, therefore provide an opportunity to investigate a company with previous software development and Web site design experience, in their development approach of Web-based applications.

The choice of Calico Commerce for this case study is based on the organisations prior extensive experience in developing Web-based applications. This presents an excellent opportunity to examine how a professional software development company currently developing Web-based applications, compares to that of companies that develop Web sites and Web-based applications.

6.3 The Target Population- A Preliminary Description

6.3.1 Primary Function

All interviewed organisations responded to the question which identified their primary function.

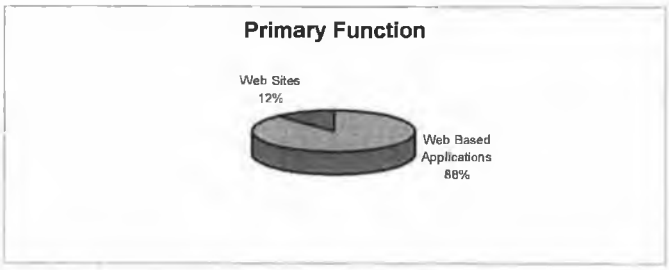


Figure 6.1: Average Primary Function

As indicated, the target companies expend an average of 12% of their effort in developing static Web sites. On the other hand, 88% of their systems development effort is applied to the development of Web-based applications. The average primary functions of the organisations are represented in figure 6.1.

NovaVision predominantly develop Web-based applications. These account for 85% of their systems development, while the remaining 15% are concerned with the development of Web sites.

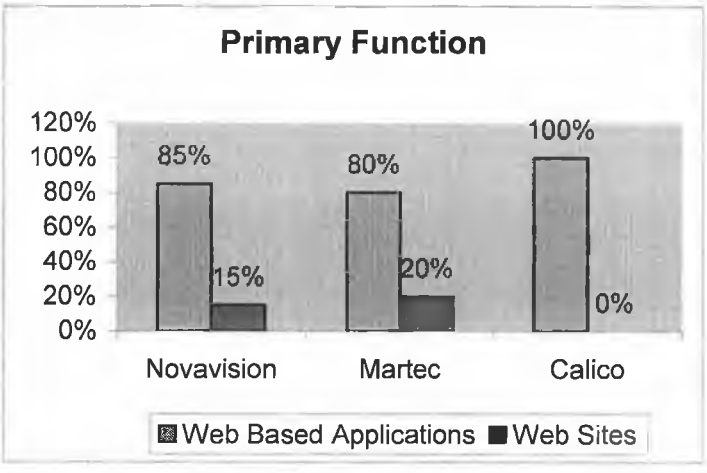


Figure 6.2: Primary Function

Martec are also predominantly developing Web-based applications. These account for 80% of Martec's development.

Their Web sites, excluding any application development account for 20% of their development. Calico Commerce, develop Web-based applications only. Figure 6.2 presents a breakdown of the primary function of each of the respondents.

6.3.2 Commencement Of Trading And Number Of Employees

All organisations came into existence between 1996 and 1998. This period saw an exponential increase in the use of the World Wide Web, as well as the potential business that the World Wide Web could facilitate. NovaVision and Martec both saw the need for Web site development companies. NovaVision opt to provide, *e-Business, Web Development and Systems Integration*, while Martec opt to provide *Web Design*, *Information Security* and *Customisable Software.* Calico Commerce offers businesses the ability to integrate configurators to their product offerings. Configuration tools assist buyers to custom-build products online. Calico Commerce realised that customers were demanding more customised products and that the idea of one-size fits all, was slowly superseded by customised products.

6.4 Findings

The following findings are presented in terms of current and future perceptions with regard to the development of Web-based applications and Web sites. The research is based on a comparison and analysis of the methodologies, techniques and tools used in the development of Web-based applications.

6.4.1 The Increase Of Web Applications

NovaVision, Martec and Calico Commerce all state that Web-based applications add a significant contribution to each of the individual companies value chain.
All three companies agree that providing Web-based applications as a service increased their potential client base. NovaVision and Martec identified an increase in the demand for Web-based applications. This is greatly influenced by the rapid expansion of the Internet. From NovaVision's perspective this is also influenced by their partnership with IBM, as well as the increase in customer requests. Martec attribute the increase of their Web-based applications, specifically to the increase of customer requests for more complex sites. They also state that projects are changing from the traditional static Web site, to more complex, graphical and interactive Web-based applications.

Calico Commerce focus all of their products around Web-based applications. Calico Commerce have two categories of customers: The first is that of Internet Based Companies, ones without traditional brick and mortar locations, and that of mid-to-large enterprises, with physical locations.

The latter use the Internet to streamline business processes.

As overall systems developed by NovaVision, the proportion of Web-based applications are *unchanging*, however Web sites are *decreasing* as a proportion of the systems developed.

With regard to Martec, Web-based applications are *increasing*, while Web sites are *decreasing* as the overall systems developed. The Calico Commerce representative stated that overall: Web sites from a commercial point of view are *decreasing*, and Web-based applications are *increasing*. The research suggests that Martec may not be attaining as many clients as that of NovaVision, for the development of Web-based applications.

6.4.2 Average Cost Of Systems Developed

All interviewees responded to the question with regard to the average cost of developing Web-based applications and Web sites. As indicated, Table 6.2 illustrates an analysis of the average cost of the latter.

NovaVision assign the majority of their development time to that of Web-based applications, as compared to that of static Web sites. They state that the average cost of a Web-based application is between €5,000 - €19,000, while the average cost of their Web site development is less than €5,000.

Martec devote most of their development time to the development of Web-based applications, however their development costs fall into the same bracket for developing Web-based applications as that of developing Web sites. Martec state that there are outliers, i.e., some Web-based application projects are priced significantly higher than the range allocated in Table 6.2. They state that the cost of developing a Web-based application and a static Web site are between €5,000 - €19,000.

The research suggests that Martec price their static Web sites significantly higher than NovaVision. This may be due to the additional services that Martec provide such as *Internet Consultancy, Corporate Branding, Information Architecture, and Internet Marketing.*

A substantial element of the service provided by Calico Commerce is associated with maintenance. This is because the business needs of the customers are dynamic and evolving. Calico Commerce develop Web-based applications. The cost of which is generally between €100,000 - €1,000,000.

Table 6.2: Average Cost Of Web-Based Applications And Web Sites

	NovaVision	**Martec**	**Calico Commerce**
Web-Based Applications	Between €5,000 - €19,000	Between €5,000 - €19,000	Between €100,000 – €1,00,000.
Web Sites	Less than €5,000	Between €5,000 - €19,000	N/A

Table 6.2 Average Cost Of Web-Based Applications And Web Sites

While the range of systems developed by Martec and NovaVision fall between €5,000 - €19,000 cost to customer bracket, those developed by Calico Commerce are substantially more expensive.

This may be attributed to the fact that Calico Commerce's customer base by far exceeds that of NovaVision and Martec. Despite the fact that NovaVision and Martec both have an international client base, their development costs are not as significant as that of Calico Commerce.

6.4.3 Customers

NovaVision, Martec and Calico Commerce all have a substantial client base. However, Calico Commerce customers are the larger players. They operate primarily in the Business to Business market sector. In the case of Calico Commerce, each business negotiates its own set of business terms and contractual relationships. The products and services provided by Calico Commerce are targeted for small-to large-size business enterprises selling complex products over the Internet, through existing channels, and within online trading communities. Their customer base include AT&T, Best Buy, Cabletron, Cisco, Dell, Eastman Kodak, edc.com, Exclaim, Gateway, Home Depot, JLG, MTD, Merrill Lynch, Motorola, Next Monet, Nortel, OneSwoop, Pro-medix, Racal, Rockwell, Siemens, Snap-on, Sunrise Medial, Telia, Telxon, Trailmobile, Tyler Refigeration, US West, UAL, Zurn Industries.

NovaVision provide solutions in the Business to Business, Business to Customer and Eprocurement market sector. Their customer base include Basta Parsons, Baxolve™ - Neurotech, Whirlpool, Helvatia Patria, Medtronic AVE Intranet, Mercedes Benz, and the Irish Medical Times Portal.

Martec's customer base include East of Boston, Career Changes, Trade2Trade, Citybin, Glasan Holiday Village, 1StopTravel, Celtic Impression and Kenny Development.

6.4.4 Development Effort

Table 6.3 is based on valid responses to the relevant questions from NovaVision, Martec and Calico Commerce. This record illustrates the percentage effort applied within the functional areas of; *project planning and management, systems analysis, system design and coding, testing and debugging new information systems* and *modifying or enhancing systems* previously developed.

Table 6.3: Development Effort

Functional Area	**NovaVision**	**Martec**	**Calico Commerce**
	% Effort	% Effort	% Effort
Project Planning And Management	7.5%	20%	10%
System Analysis	7.5%	10%	10%
System Design And Coding	70%	50%	50%
Testing And Debugging New Information Systems	7.5%	10%	10%
Modifying Or Enhancing Systems Previously Used.	7.5%	10%	20%

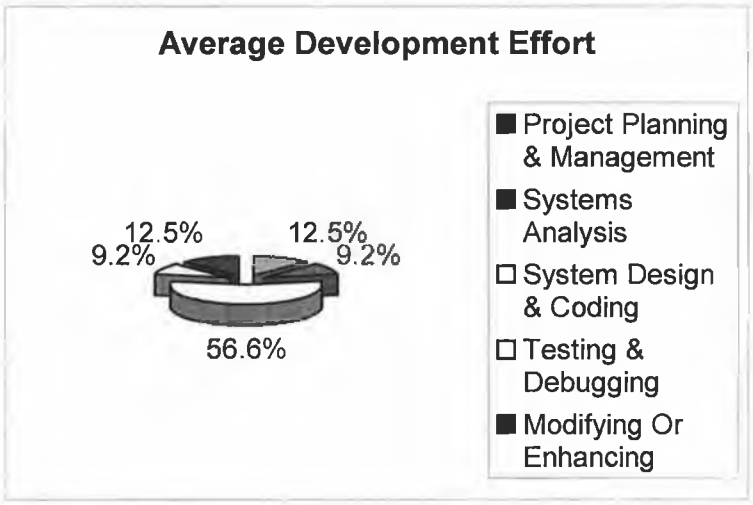
Table 6.3 Development Effort

NovaVision, Martec and Calico Commerce all allocate most of their development effort to Systems Design and Coding. NovaVision allocate 70% of their development effort, Martec allocate 50% of their development effort and Calico Commerce allocate 50% of their development effort to systems design and coding. All three companies feel that it was necessary to allocate most of their development effort, simply because it is the core of the project.

There is a parallel between Martec and Calico Commerce with regard to their development effort in terms of systems design and coding. Both companies allocate 50% of their development time to this area. The research suggests that this may be attributed to the applications that both companies are using in their development. The applications that each company utilises will be discussed later in the chapter.

Figure 6.3:Average Development Effort

All respondents have the highest concentration of staff with responsibility for system design and coding.



This reduces marginally through project planning, modifying systems, systems analysis and testing and debugging. Figure 6.3 identifies the average development effort by all three respondents.

6.4.5 Complexity Of Web-based Applications & Web Sites

All of the companies were asked to rate the following on a scale of one to five:

- The functional complexity of the average Web-based application and Web site developed by them.
- The data structure complexity of the average Web-based application and Web site developed by them.
- The user interface complexity of the average Web-based application and Web site developed by them.

The results are presented in Table 6.4. All three companies rate functional complexity highest, data structures complexity and user interface complexity next. The overall average complexity of Web-based applications is 4.3, and the overall average complexity of Web sites is and 2.4.

Table 6.4: Complexities Of Web-based Applications

	NovaVision		Martec		Calico Commerce	
Categories	**Web-based Applications**	**Web Sites**	**Web-based Applications**	**Web Sites**	**Web-based Applications**	**Web Sites**
Average Rate for Functional Complexity	5 (extremely complex)	2 (minimal)	4 (complex)	3	5	
Average Rate for Data Structure Complexity	5 (extremely complex)	2 (minimal)	3.5 (quite complex)	2.5	5	
Average Rate for User Interfaces Complexity	4 (complex)	2 (minimal)	4.5	3	3	

Table 6.4 Complexities Of Web-based Applications

Web-based applications are complex by nature. This is identified in the research, where all three organisations rate functional complexity the highest. This is not a surprise, as Web-based applications are powerful tools that present and deliver information in a variety of media. The applications that each company are using attributes to the complexity of the user interfaces in terms of development. NovaVision, Martec and Calico Commerce all use Macromedia Dreamweaver in the development of Web-based applications.

The research indicates that despite the fact that all three companies are using a widely renowned development application, their user interface development is nevertheless a complex process.

Martec state that e-commerce facilities attribute towards the complexity of Web based applications. *"With Web based applications there is a complexity at the frontend, such as search and calculations" (David Martin, Martec).*

NovaVision attribute complexity of Web based applications specifically to one of their development applications. *"Webshpere is a very complex user interface. There is no way around it, it is just extremely complex" (Shane McGinley, NovaVision).*

6.4.6 Methodologies: Past And Current Use

This section reports the findings on the past, and current use of development methodologies to develop Web-based applications and Web sites.

From the research conducted it is concluded that NovaVision, Martec and Calico Commerce have ***not previously*** or do ***not currently*** use any formal Web specific development methodology in the design and development of Web-based applications or Web sites. NovaVision, Martec and Calico Commerce do not adhere to any specific Web application development methodologies that have been put forward by leading authors in that field.

Guidelines and tools are in place to assist the developer, however all of the companies follow their own individual in-house approach. Following is a discussion of the In-House Methodologies applied by NovaVision, Martec and Calico Commerce.

6.4.6.1 NovaVision: In-House Methodology

NovaVision use object oriented methods in the development of Web-based applications. However, they do not use a Web specific object oriented methodology. They follow an in house methodology for the development of Web sites. They state that they use object-oriented methods, primarily because of its re-usability.

They feel that object-oriented methods are advantageous, and do not possess any major weakness. This is the approach that NovaVision have taken in the **past**, and are **currently** utilising.

They do not feel the need to move to any new development methods. Research suggests that there may be some weaknesses associated with the methods that they are using, however in terms of quality, they may not be affected. However, there are weaknesses associated with object oriented methods, NovaVision do not encounter them. *"We feel that there are really no weaknesses" (Shane McGingley, NovaVision).* See section 3.16 for weakness of object oriented methods.

NovaVision state that their development methodology facilitates the entire development lifecycle including *systems analysis*, *development*, *testing* and *delivery.* They adhere to the following guidelines when developing a project:

- *Usability:* Develop Web-based applications and Web sites that are easy to use.
- *Learnability:* Use of the site should be intuitive and self explanatory.
- *Cross Brower Capability:* Web-based applications/Web sites are capable of being accessed from a wide range of browsers.
- *Compression of all graphics:* Reduce bandwidth requirements.
- *Metadata:* Ensure pages are categorised. To help ensure registration with search engines.
- *Reduce Download Time:* Applications are designed to facilitate rapid response.

At the end of production NovaVision provide full documentation in relation to the Website design and Web-based application. This includes:

- Written procedures to be followed in the event of failure of the site.
- Defined procedures for maintaining and updating the site.
- Copies of all images, style sheets and templates used.

NovaVision employ a contract Web designer, therefore not all of their services remain in-house. They partner with companies that provide technology, such as IBM, Inkotmi, Cityline, Novanet, Customer Focus Company, and Eglogal Solutions. Each of the above companies supplement specialised capabilities that NovaVision apply to individual projects.

They pride themselves on "*good design*" based on the appropriate use of colour, proportion, typeface and functionality.

6.4.6.2 Martec: In-House Methodology

Martec use an in-house methodology for both the development of their Web-based applications and Web sites. They have developed their methodology typically from good practice. The majority of their Web site development has transpired from analysing what others have developed. This involves assessing the quality of the Web application in terms of usability.

Martec state that they look at other e-commerce sites, such as Amazon.com. They develop their methodology from that, in terms of layout, and structure. This is the method they are **currently** using and have used in the **past.** Martec have their own procedures and policies in terms of developing systems. All of their policies have evolved over time. They state that their development methodology returns control of the development process to their clients, and at all stages of the process their clients are informed. They do not employ contractors, so therefore all of their services remain in-house.

Martec follow a process that encompasses the following:

1. *Define the Task:* They seek to understand the customer's requirements and then advice the customer on the best solution and provide the cost of the project.
2. *Create the solution:* They prototype, design, test and launch the solution within strict quality guidelines and delivery schedule.
3. *Support the client:* They offer training, content management, maintenance systems and support.

Martec follow a quality control system when they get into the system development life cycle. They have a 31 item checklist that encompasses a series of checks.

They complete their checks at every stage of the design and development process. Following that, they convert their prototype design into a Web page template and run a series of tests to ensure that it is compatible with different monitor screen sizes and browsers.

6.4.6.3 Calico Commerce: In-House Methodology

Calico Commerce do not use any specific methodology. Their development methods involve:

- *Information Gathering:* Obtaining all of the requirements from the customers.
- *Identifying Tasks and Phases:* Breaking the project down in term of the tasks that need to be completed in development.
- *Develop Schedule:* Estimating the length of the project.
- *Time Management:* Entering all project details into Microsoft Project.

Calico Commerce states that their customers very much underestimate the importance of accurate planning and the use of a proper methodology to achieve their goals. From the point of view of their customers, it is more a case of *"how long do you think it will take to get the project completed, not a case of what methodology is going to be used?"*. In terms of estimation and costs, they are not included in any methodology.

6.4.7 Methodologies: Planned Usage

NovaVision are very satisfied with the approach that they are currently using. The Object-Oriented approach has worked very well in the past and they intend to continue to use an object-oriented approach. NovaVision state that the object-oriented approaches that they are using, are completely free from any weaknesses. Therefore, they do not feel any need to move or to adopt any other approaches when designing Web-based applications or Web sites. Despite that, NovaVision do not intend to adopt a Web specific object oriented development methodology, such as OOHDM (Object Oriented Hypermedia Design Methodology).

Martec intend to continue with the development process that they are currently using. They develop their methodologies typically from good practice. They analyse other sites, looking at quality in terms of usability, layout and structure. Martec mentioned Amazon.com as one of the sites that they would analyse. Their development methodology has evolved over time. Martec do not have any intention to change the current in-house process that they have in place.

Calico Commerce intend to continue with the method that they are currently using, despite the fact that it lacks any planning and strategy. They will continue to develop Web-based applications following the non-structured development approach.

6.4.8 Development Tools

This section reports the findings on the use of development tools, scripting languages and databases in the development of Web-based applications and Web sites.

6.4.8.1 NovaVision: Development Tools

NovaVision state that their primary development tool is Websphere, which they use for Java Projects. Webshpere accounts for 60% of their development. Their other most widely used development tools include Microsoft InterDev, which accounts for 25%, Macromedia Dreamweaver, which accounts for 10%, Adobe Photoshop, which accounts for 4% and Text Pad, which accounts for 1%. See Table 6.5

Table 6.5: NovaVision: Development Tools

Web Layer	
Development Application Tool	***Usage Percentage***
Websphere *(Primary Development Tool)*	60%
Microsoft Visual InterDev	25%
Macromedia Dreamweaver	10%
Adobe Photoshop	4%
TextPad	1%
Total	***100%***

Table 6.5 NovaVision: Development Tools

NovaVision are a partner with IBM. Websphere is an IBM product. Therefore this coincides with the fact that they use Websphere for 60% of their development. IBM's Websphere Studio Application Developer 5, is a software package that includes a Webserver and an Application server.

In NovaVisions case it also facilitates easy connection to IBM's DB2 Database. NovaVision state that if they are developing a Java based project they would consequently use Websphere for their development. However if they were developing an ASP project, they would in turn use Microsoft Visual InterDev. They attribute the use of Microsoft InterDev to the extensive Microsoft libraries that accompany it, as well as the excellent code formatting facilities that it attains. Their choice is deciding to use Phototshop and Dreamweaver was simply due to the fact that they are easy to use, and that they are excellent products.

NovaVision state that there are some slight differences in the applications used in the development of Web-based applications and Web sites. They state that Java is strong, stable and multi-platform, this being its huge advantage. However, they state that Java can be slow, therefore identifying a weakness of the tool.

6.4.8.2 Martec: Development Tools

The following is an account of the development tools, scripting language and databases used by Martec for the development of Web-based applications and Web sites. There is not a significant difference in the applications that Martec use in the development of Web-based applications and Web sites. If Web sites have the complexity of the Web-based applications, they use exactly the same tools.

Martec state that their primary development tool is Dreamweaver. The latter accounts for 30% of their development. Their other most widely used development tools include Macromedia Fireworks, which accounts for 20%, Microsoft Access, which accounts for 20%, Macromedia Flash, which accounts for 20%, and MS SQL, which accounts for 10%. See Table 6.6

Table 6.6: Martec: Development Tools

Web Layer	
Development Application Tool	***Usage Percentage***
Macromedia Dreamweaver *(Primary Development Tool)*	30%
Macromedia Fireworks	20%
Flash	20%
Total	***70%***
Database	
Database	***Usage Percentage***
Microsoft Access	20%
MS SQL Server	10%
Total	***30%***
Total	***30%***

Table 6.6 Martec: Development Tools

Martec state that the reason they use Dreamweaver as their primary development tool is simply due to its functionality. Martec uses Macromedia Fireworks, Microsoft Access, Flash, and MS SQL because of their compatibility and ease of use.

6.4.8.3 Calico Commerce: Development Tools

Calico Commerce identified the following used in the development of Web-based applications. See Table 6.7.

Table 6.7: Calico Commerce: Development Tools

Web Layer	
Application	***Usage Percentage***
Macromedia Dreamweaver (Primary Development Tool)	40%
Visual InterDev	10%
Total	**50%**
Databases	
Application (Databases)	***Usage Percentage***
Oracle	35%
Microsoft Access	12.5%
MS SQL Server	2.5%
Total	**50%**

Table 6.7 Calico Commerce: Development Tools

Calico Commerce states that their primary development tool is Dreamweaver. It accounts for 40% of their development. Their other most widely used development tools include Microsoft Visual InterDev which accounts for 10%, Oracle which accounts for 35%, Microsoft Access which accounts for 12.5% and MS SQL Server, which accounts for 2.5%.

6.4.9 Comparison Of Primary Development Tools

Table 6.8 identifies the primary development tools used by each of the three subject organisations.

Table 6.8: Comparison Of Applications			
Development Tool/Database	***NovaVision***	***Martec***	***Calico Commerce***
Access		20%	12.5%
Dreamweaver	10%	30%	40%
Fireworks		20%	
Flash		20%	
IBM Websphere	60%		
MS SQL Server		10%	2.5%
Oracle			35%
Photoshop	4%		
Text Pad	1%		
Visual InterDev	25%		10%

Table 6.8 Comparison Of Applications

Table 6.8 suggests that Macromedia Dreamweaver is by far the most widely used development application tool used by NovaVision, Martec and Calico Commerce. Dreamweaver is used by all three companies for their Web layer development. Dreamweaver is suitable for HCI design.

Microsoft Visual InterDev proves to be another popular tool for Web layer development as both NovaVision and Calico Commerce opt to use this application. In terms of Databases, Microsoft Access and MS SQL Server are both used by Martec and Calico Commerce. From the research conducted these prove to be the most widely used databases in terms of developing Web-based applications.

6.4.10 Development Tools: Future Use

NovaVision state that they have researched the .NET platform. However, they decided against it as ASP is enabling them to complete everything that they are required to do. NovaVision also state that ASP is a technology that they are familiar with. They state that ASP code is reusable, this being an additional bonus. Taking all of the above into consideration, NovaVision intend to use the current tools, languages and databases that they have in place.

Due to the fact that NovaVision use Websphere for 60% of their development, they admitted that if Websphere were to upgrade or include additional features, then obviously they would move to the upgraded version and educate themselves of any additional features that would accompany the new version.

Martec state that they are currently looking at the .NET platform. Due to customer demand Martec are compelled to develop in PHP and MYSQL in the future.

Martec are using XML, however in the future they intend to use it more extensively. In terms of industry standards, XML is fast becoming the standard for data representation.

Calico Commerce did go through a technology transition, in that they moved from Microsoft based technologies too Java. Calico Commerce felt that they were limited with the original configuration engine. They moved too Java due to the fact that they could extend it. Calico Commerce can add to a Java application whenever required. The user interface was not tied into the engine, so there was a separation of content and functionality

6.4.11 Formal Development Techniques

This section reports the findings on the use of formal development techniques in the development of Web-based applications and Web sites. Formal development techniques are widely adopted by all respondents, with NovaVision, Martec and Calico Commerce using them in their design process. Table 6.9 presents the development techniques used by respondent organisations.

Table 6.9: Development Techniques						
Techniques	***NovaVision***		***Martec***		***Calico Commerce***	
	Web-Based Applications	***Web Sites***	***Web-Based Applications***	***Web Sites***	***Web-Based Applications***	***Web Sites***
Action Diagrams					✔	
Data Flow Diagrams	✔		✔		✔	
Decision Tables			✔			
Decision Trees						
Entity Life Histories						
Entity Relationship Diagrams			✔			
Normalisation			✔			
OO Techniques						
Program Flowcharts			✔			
Pseudocode	✔				✔	
Scenarios			✔			
Structured English						
System Flowcharts			✔			
Warnier-Orr Diagrams						
UML						
Other Please Specify					Story boarding	

Table 6.9 Development Techniques

NovaVision identified the use of Data Flow Diagrams for the purpose of "*specking out a project*". They stated that they would regard Program Flow Charts, and System Flowcharts similar to that of Data Flow Diagrams. Pseudocode is a technique that they would also use. They state that all of the above techniques are used in Web-based application development both ***currently*** and in the ***past***.

NovaVision also state that Program Flow Charts are a technique that they use in the development in Web sites, both ***currently*** and in the ***past.***

Martec singled out the use of Data Flow Diagrams, Decision Tables, Entity Relationship Diagrams, Normalisation, Program Flow Charts, Scenarios, System Flowcharts and Storyboarding in the development of Web-based applications.

They state that due to the fact that Web design is very visual, the above tools are very beneficial in the organisation of code. In terms of database structure, Martec regard Entity-Relationship Diagrams as the best, as they are geared towards database design.

Calico Commerce identifies Action Diagrams, Data Flow Diagrams, Psuedocode and Story Boarding as formal techniques that they use in the development of Web-based applications.

The research suggests that Martec use the most formalised techniques in the development of Web-based applications and Web sites. This therefore suggests that they apply more planning to the design of their Web applications.

6.4.12 Formal Development Techniques: Future Use

NovaVision state that they are satisfied with the development techniques that they are ***currently*** using and have ***previously*** used. They do not intend to use any other techniques, other than if they are forced to, by a change in the technology. Martec also state that they are satisfied with the development techniques that they are currently using or previously used. They do not have any intention to use any other techniques.

However, if a specific technique is requested by a customer that they have not previously used, then of course they will adopt it. Calico Commerce will continue using the exact development techniques, and do not intend to adopt any others, unless requested.

6.4.13 Summary Of In-House Methodologies

The following section provides a summary for each of the tools, techniques and methods used by NovaVision, Martec and Calico Commerce.

6.4.13.1 NovaVision

The software process model used by NovaVision follows a similar process to that of the System Development Life Cycle (SDLC).

The software process as mentioned in the table below, combined with the techniques that NovaVision use suggests that they may be following the Hypermedia Flexible Process Modelling (HFPM) approach (Olsina, 1998). See Table 6.10.

Table 6.10: Summary Of In-House Methodology - NovaVision

NovaVision *(Summary of In-House Methodology, Tools and Techniques)*		
Methodology	***Tools***	***Techniques***
Similar to the System Development Life Cycle	Websphere	Object-Oriented Methods
	Visual InterDev	Data Flow Diagram
	Dreamweaver	Pseudocode
	Photoshop	
	Text Pad	

Table 6.10: Summary Of In-House Methodology - NovaVision

The Hypermedia Flexible Process Modelling (HFPM) approach presented by Olsina (1998) is an engineering based approach. HFPM embraces functional, methodological, informational, behavioural, and organisational views of the hypermedia development process. The Hypermedia Flexible Process Modelling (HFPM) covers all essential phases and activities of a hypermedia project; it assists in establishing milestones and methods as well as promoting communication and human understanding.

6.4.13.2 Martec

Martec state that a specific methodology is not used in the development of Web-based applications. They concede that they may be using a specific methodology, but may not be aware of it. The approach taken may be described as ad-hoc.

The strategy focuses around the analysis of other E-commerce sites in term of usability. They state that the methods that they are using do in fact cover the entire life cycle. This combined with the techniques that they apply suggests that Martec may be following the Lowe-Hall's Engineering Approach (Lowe et al., (1999). See Table 6.11

Table 6.11: Summary Of In-House Methodology - Martec

Martec *(Summary of In-House Methodology, Tools and Techniques)*		
Methodology	***Tools***	***Techniques***
Ad-hoc	**Dreamweaver**	Data Flow Diagrams
	Fireworks	Decision Tables
	Access	Entity Relationship Diagrams
	Flash	Normalisation
	MS SQL Server	Program Flow Charts
		Scenarios
		System Flow Charts
		Story Boarding

Table 6.11: Summary Of In-House Methodology - Martec

Lowe et al., (1999) provide a framework for the development of hypermedia applications. The framework includes domain analysis, product modelling, process modelling, project modelling, development and documentation. The activities that are performed during the development process are: systems analysis, design, production, verification, testing and maintenance.

6.4.13.3 Calico Commerce

The software process model used in the development of Calico Commerce followed a similar process to that of the System Development Life Cycle.

The developer's, however are not aware that they were following any process, but rather, they are conducting the development using *"common senses"* within a certain timeframe using whatever means possible to complete the project. A specific methodology is not used in the development of Web applications. The approach may be described as ad-hoc, and consists with a strategy that Calico Commerce considers to be the core competency. The strategy focuses around the use of a senior software developer.

The philosophy adopted during the design and construction was a delivery of *"design"* and *"content"* within a certain timeframe. *"It is not a matter of how we develop our applications, but more a factor of how quickly can we deliver them to our customers" (David Heffernan, Calico).* However, customers do expect the highest level of quality.

Calico Commerce's strategy uses the latest technology in the creation of Web applications. Technology is therefore a key driver in the design and development of Calico Commerce's Web applications. See Table 6.12.

Table 6.12: Summary Of In-House Methodology - Calico Commerce

Calico Commerce *(Summary of In-House Methodology, Tools and Techniques)*		
Methodology	***Tools***	***Techniques***
Similar to the System Development Life Cycle	Access	Action Diagrams
	ASP	Data Flow Diagrams
	HTML	Pseudocode
	MS SQL	
	Oracle	
	XML	

Table 6.12: Summary Of In-House Methodology - Calico Commerce

After the case studies were completed, it was decided to conduct a telephone survey. The purpose of the latter being to determine whether or not the finding of the research could be extrapolated to a wider development community.

This survey took a random sample of 35 Web Development companies throughout Ireland from the Software Directorate. Each company were asked the following questions:

(1) Are you aware of any Web specific development methodologies for the development of Web-based applications?

Only 5.7% of the respondents were aware that Web specific development methodologies actually exist.

(2) *Do you use a Web specific development methodology for the development of Web-based applications?*

It was concluded from the telephone survey that 97.1% of the respondents do not actually adhere to a Web specific development methodology when developing Web-based applications.

The telephone survey solidified the research findings by clarifying the following: The majority of companies were unable to identify a Web specific development methodology, and

- The universally available Web specific development methodologies are virtually never used for the development of Web applications.

6.5 Summary And Conclusions

This section presents a summary of the case studies. Subsequent to a descriptive review of the case studies, the findings of the research are summarised and the conclusions are presented. Finally, suggestions for further research are presented.

6.5.1 Summary Of The Findings

NovaVision, Martec and Calico Commerce all state that Web-based applications add a significant contribution to each of the individual companies value chain.

All three companies agree that providing Web-based applications as a service increases their potential client base. NovaVision and Martec identified an increase in demand of Web-based applications.

The range of systems developed by NovaVision and Martec fall between €5,000 - €19,000 cost to customer bracket. The cost of the latter for Calico Commerce is generally between €100,000 and €1,000,000. Those developed by Calico Commerce are substantially more expensive. This may be attributed to the fact that Calico Commerce customer base, by far exceeds that of NovaVision and Martec.

NovaVision operate primarily in the Business to Business, Business to Customer and Eprocurement market sector. Calico Commerce operate primarily in the Business to Business market sector.

NovaVision, Martec and Calico Commerce all apportion most of their development effort to Systems Design and coding. There is a parallel between Martec and Calico Commerce with regard to their development effort in terms of system design and coding. Both companies allocate 50% of their development time to this area. The research suggests that this may be attributed to the applications that both companies are using in their development.

All three companies rate functional complexity highest. The overall average complexity of a Web-based application is 4.3, and the overall average complexity of Web sites is 2.4. All three organisations use Dreamweaver when designing the user interface.

NovaVision, Martec and Calico Commerce have not previously or do not currently use any formal Web specific development methodology in the design and development of Web-based applications or Web sites. They do not intend to adopt any Web specific development methodology in the future.

Webshpere is the primary development tool used by NovaVision. It accounts for 60% of their development. Dreamweaver is the primary development tool used by Martec for their Web layer development. Dreamweaver accounts for 30% of their development, while Microsoft Access accounts for 20% of their database development. Dreamweaver is the primary development tool used by Calico Commerce. Dreamweaver accounts for 40% of their Web layer development, while Oracle accounts for 35% of their database development. NovaVision and Calico commerce use Visual InterDev in their Web layer development.

In terms of database design, Microsoft Access and MS SQL server are both used by Martec and Calico Commerce. From the research conducted these prove to be the most widely used databases in terms of developing Web-based applications.

Formal development techniques applied by NovaVision include Data Flow Diagrams, System Flowcharts, Pseudocode and Program Flow Charts.

Techniques applied by Martec include Data Flow Diagrams, Decision Tables, Entity-Relationship Diagrams, Normalisation, Program Flow Charts, Scenarios, System Flowcharts and Storyboarding. Calico Commerce identifies Action Diagrams, Data Flow Diagrams, Pseudocode and Story Boarding as the formal techniques used in the development of Web-based applications.

The software process combined with the techniques that NovaVision use suggests that they may be following the Hypermedia Flexible Process Modelling (HFPM) approach (Olsina, 1998). The approach used by Martec, combined with the techniques that they use, suggest that they may be unknowingly following the Lowe-Hall's Engineering Approach (Lowe et al., 1999).

6.5.2 Conclusions

Hypermedia applications for the Web are mostly developed ad hoc, progressing usually from small too large applications and becoming uncontrollable. Guidelines and tools are beginning to appear assisting the hypermedia developer. Current practices miscarry, due to scarcity of planning, and inappropriate techniques, process and methodologies (Kock, 2002).

The research indicates that the consensus for Web specific methodologies has not reached general acceptance by Web developers. The perceptions of its usefulness, and projected reception into the development process have a long way to go, before it is widely accepted by Web professionals. With regard to NovaVision, Martec and Calico Commerce the following conclusions are drawn:

- The vastly published methodologies are not used at all.
- They develop according to in-house methodologies based on what suits them, and what has worked in the past. The rationality for this development of in-house methodologies includes preference for adapting specific traditional methods to suit a specific task.

The research suggest, expectations for high levels of usage of Web specific methodologies is further reduced by the following:

- Only one of the respondents were capable of identifying some of the Web-based application development methodologies.
- One of the respondents admitted that perhaps the methods that they are following falls into the category of a Web specific methodology, however they would not be able to identify which one.

The data suggests however, that unknown to themselves, that NovaVision and Martec are applying pockets of Web specific methodologies to their application design. None of the interviewees are applying a comprehensive Web specific methodology to Web-based application design.

The little use of published methodologies is not a result of inadequate training in system development methodologies but rather that professional developers prefer their own customised methodologies. All interviewees state that in-house methodologies tend to facilitate each customised project more effectively and efficiently than specific published methodologies, which sometimes are difficult and irritating from a developer's perspective. "*Traditional development methodologies and software engineering methods are unsuited to hypermedia systems*" (Lang, 2001).

In contrast to this, the research suggests that sections of traditional methodologies are currently being adjusted by organisations to fit the needs of the Web environment. The analysis of the development of Web-based applications within an organisational context has revealed numerous issues resulting from *technology*, *organisational methods* and *customer requests*.

Web-based application development has been characterised as *"guerrilla programming in a hostile environment using unproven tools, processes and technology"* (Thomas, 1998).

For organisations, embracing Web specific methodologies constitutes a high-risk venture. The interviewees remain unconvinced that there is an advantage from adopting Web specific methodologies. This therefore suggests that high levels of usage are unlikely. It is probable that neither NovaVision, Martec nor Calico Commerce will adopt a Web specific methodology in the next three years.

The analysis of NovaVision, Martec and Calico Commerce has illuminated additional areas of concern that were not initially the focus of the study. The organisational circumstances of the three case studies presented a challenge to inspect any deficiencies that may exist in the development process of Web applications.

The study of NovaVision, Martec and Calico Commerce, identified that deficiencies exist in the development of Web applications, especially in the areas of *application technology, procedures/guidelines* and *documentation.*

In the case of NovaVision and Martec, technology is identified as a potential problem in the design of Web applications. Improved technology is arriving on the market at a speedy pace. This poses many questions for NovaVision and Martec so far as they must decide whether to stay with the application they are using, or move too something completely new. Companies may be forced by competition to increase productivity.

Web application developers may tend to design for the present, due to the fact that technology is changing so quickly. Designing for future applications may be reserved, as developers are focused on ensuring the system will meet its objectives, and be delivered within budget and time restraints. This may be a priority over technological advances that may occur in the future.

In the case of Calico Commerce, it appears that the time restraints that developers face during development may lead directly to an absence of documentation. The timeframes in which Calico Commerce are required to develop their systems may lead to the demise of any procedures or guidelines being followed.

When summing up, all interviewees feel the issue of Web specific methodologies is over-hyped and expressed doubt over the academic papers put forward by Web Engineers. This leads to the question*: How will Web developer's deal with future Web-based applications, which will encompass greater complexity and functionality?*

6.5 Suggestions For Further Research

This thesis constitutes a platform study and as such is descriptive and exploratory by nature. As the research progressed, several areas deserving more focused investigation surfaced. In particular the low level of usage of Web specific development methodologies. As can be seen from Section 6.4.6. neither NovaVision, Martec nor Calico Commerce use a Web specific development methodology. All interviewees struggled when questioned regarding their methodology. **The levels of awareness of universal Web specific methodologies among Web developers needs further qualification.** More definite research in this area may be more productive if taken from a broader angle.

Further work on developing specific tools and techniques that can be combined to provide customised in-house methodologies may prove to be useful to the Web development profession. **In order for methodologies, techniques and tools to evolve in such technologically dynamic environments, research into the present development practices must continue.**

BIBLIOGRAPHY

ABELSE, EILEEN, G., & DOMAS WHITE, MARILYN, & HAHN, KARLA, (1998)
"A User-Based Design Process For Web Sites.": INTERNET RESEARCH: ELECTRONIC NETWORKING APPLICATIONS AND POLICY, VOL. 8, NO. 1, 1998. MCB UNIVERSITY PRESS.

ALLEN, C., PAUL, (1991)
"Effective Structured Techniques: From Strategy To Case.": PRENTICE HALL, LONDON.

ALTER, STEVEN, (1992)
"Information Systems: A Management Perspective.": ADDISON-WESLEY PUBLISHING COMPANY.

ANDERSON, RON, (2001)
"The Long And Winding Road To Web-Based Apps.": NETWORK COMPUTING, MARCH, 19, 2001.

ANTHONY, ROBERT, (1965)
"Planning And Control Systems: A Framework For Analysis.": CAMBRIDGE: HARVARD UNIVERSITY GRADUATE SCHOOL OF BUSINESS STUDIES ADMINISTRATION.

ARTZ, JOHN, M., (1996)
"A Top-Down Methodology For Building Corporate Web Applications.": INTERNET RESEARCH:ELECTRONIC NETWORKING APPLICATIONS AND POLICY, VOL. 6, NO. 2/3, 1996.

ASHWORTH, C., & GOODLAND, M., (1990)
"SSADM - A Practical Approach.": MCGRAW-HILL BOOK COMPANY, LONDON.

AVISON D.E., & SHAH, H.U., (1998)
"The Information Systems Development Cycle.": A First Course In Information Systems.": MCGRAW HILL.

AVISON, D.E., & FITZGERALD, G., (1995)
"Information System Development: Methodologies, Techniques And Tools." : Second ed., MCGRAW HILL, LONDON.

BEAUMONT, ROBIN, (2003)
"Developing Information Systems-Getting The Users Involved.": AVAILABLE ONLINE: http://www.robin-beaumont.co.uk/rbeaumont/virtualclassroom/chap12/s4/des2.pdf

BENEDETTI, WENDELL, (2002)
"Automated Web Development Tools.": PETERSENS' PHOTOGRAPHIC, VOL. 31, NO.6,OCTOBER 2002.

BENJAMIN, R.I., (1971)
"Control Of The Information System Development Cycle.": NEW YORK, WILEY-INTERSCIENCE.

BENYON, DAVID, (1997)
"Information And Data Modelling.": Second ed., .MCGRAW HILL.

BOCIJ, PAUL, &, CHAFFEY, DAVE, &, GREASLEY, ANDREW, &, HICKEY, SIMON (1999)
"Business Information Systems: Technology, Development and Management.": FINACIAL TIMES MANAGEMENT, LONDON.

BRITTON, CAROL, & DOAKE, JILL, (1997)
"Software Systems Development: A Gentle Introduction." :MCGRAW-HILL, LONDON.

BRONZITE, MICHAEL, (2000)
"System Development: A Strategic Framework." :SPRINGER.

BROWN, DAVID WILLIAM, (2002)
"An Introduction To Object-Oriented Analysis: Objects And UML In Plain English.": Second ed., JOHN WILEY & SONS, INC.

BURDMAN, JESSICA, (1999)
"Collaborative Web Development: Strategies And Best Practices For Web Teams.": ADDISON-WESLEY.

CAINE, S.H., & GORDON, E.K., (1975)
"PDL-A Tool For Software Design.": PROCEEDINGS OF THE 1975 NATIONAL COMPUTER CONFERENCE, VOL. 44 MONTVALE, N.J. AFIPS PRESS.

CALLAWAY, ERIN, (1997)
"Method From The Madness.": PC WEEK, VOL. 1, NO.5, FEBRUARY, 1997.

CARMICHAEL, A., (1994)
"Object Development Methods.": VARIOUS CONTRIBUTORS EDITED BY CARMICHAEL SIGS BOOKS, NEW YORK.

CHEN, JIM, Q., & HEATH, RICHARD, (2001)
"Building Web Applications: Challenges, Architectures, And Methods.": INFORMATION SYSTEMS MANAGEMENT QUARTERLY, VOL 18.

CHEN, PETER PIN-SHAN, (1976)
"The Entity-Relationship Model - Towards A Unified View Of Data.": ACM TRANSACTIONS ON DATABASE SYSTEM VOL. 1, NO. 1, MARCH 1976 P9-36.

CLARK, RAYMOND, T., (1986)
"Contemporary Systems Analysis And Design.": WADSWORTH.

CLEARY, TIMOTHY, (1998)
"Business Information Technology.": PITMAN, LONDON.

COAD, PETER, & YOURDON, EDWARD, (1991a)
"Object-Oriented Design.": YOURDON PRESS.

COAD, PETER, & YOURDON, EDWARD, (1991b)
"Object-Oriented Analysis.": Second ed., YOURDON PRESS.

CODD, E.F., (1970)
"A Relational Model Of Data For Large Shared Data Banks.": COMMUNICATIONS OF THE ACM.

COMER, E., DOUGLAS, (1997)
"Computer Networks And Internet: International Edition.": PRENTICE HALL.

CONALLEN, JIM, (1999)
"Web Application Architectures.": COMMUNICATIONS OF THE ACM, VOL. 42, NO. 10, OCTOBER 1999.

COSTAGLIOLA, GENNARO, & FERRUCCI, FILOMENA, & FRANCESE, RITA, (2002)
"*Web Engineering: Models And Methodologies For The Design of Hypermedia Applications.": AVAILABLE ONLINE ftp://cs.pitt.edu/chang/handbook/58b.pdf [2002, JULY, 19].*

COX, BRAD, J., (1987)
"Object-Oriented Programming: An Evolutionary Approach.": ADDISON WESELY.

CURTIS, GRAHAM, &, COBHAM, DAVID, (2002)
"Business Information Systems, Analysis, Design And Practice.": Fourth ed., PEARSON EDUCATION LIMITED.

CURTIS, GRAHAM, (1995)
"Business Information Systems: Analysis, Design And Practice." : Second ed., ADDISION-WESLEY, WOKINGHAM.

CUTTER CONSORTIUM, (2000)
"RESEARCH BRIEFS." : 7TH NOVEMBER

DATE, C.J., (1995)
"An Introduction To Database Systems.": ADDISON-WESLEY.

DE TROYER, O.M.F., & LEUNE, C.J., (1998)
"WSDM: A User Centered Design Method For Web Sites.": AVAILABLE ONLINE http://infolab.kub.nl/prj/past/wsdm/www7/final/paper.html [2003, JANUARY, 13].

DE TROYER, OLGA, (1998)
"Designing Well-Structured Websites: Lessons To Be Learned From Database Schema Methodology.": AVAILABLE ONLINE http://infolab.kub.nl/prj/past/wsdm/psfiles/er060.pdf [2003 MARCH, 17].

DE WEGER, MARK, K., & FRANKEN, HENRY, M., & VISSERS, CHRIS, A., (1995)
"A Development Model For Distributed Information Systems."; AVAILABLE ONLINE http://home.planet.nl/~mark.deweger/Publications/1995-idc.pdf [2002, JUNE, 10].

DEMARCO, TOM, & YOURDON, EDWARD, (1978)
"Structured Analysis And System Specification." : PRENTICE HALL, ENGLEWOOD CLIFFS.

DEMARCO, TOM, (1979)
"Structured Analysis And System Specification.": PRENTICE HALL.

DENSCOMBE, MARTYN (2003)
"The Good Research Guide, For Small-Scale Social Research Projects.": OPEN UNIVERSITY PRESS.

DESHPANDE, YOGESH, & MURUGESAN, SAN, & GINIGE, ATHULA, & HANSEN, STEVE, & SCHWABE, DANIEL, & GAEDKE, MARTIN, & WHITE, BEBO (2002)
"Web Engineering.": JOURNAL OF WEB ENGINEERING, VOL. 1, NO.1, RINTON PRESS.

DÍAZ, ALICIA, & ISAKOWITZ, TOMÁS, & MAIORANA, VANESA, & GILABERT, GABRIEL, (1995)
"RMC: A Tool To Design WWW Applications.": INTERNATIONAL WORLD WIDE WEB CONFERENCE, 1995.
AVAILABLE ONLINE , http://www.w3j.com/1/isakowitz.187/paper/187.html> [2002, APRIL, 18].

DOWLING, NICK, (1998)
"Database Design And Management Using Access.": ASHFORD COLOUR PRESS, GOSPORT.

DRUDIS, ANTONI, & MENDONCA, JOHN, (2001)
"The Architecture Of Web Applications.": ENTERPRISE SOLUTIONS. AVAILABLE ONLINE. http://www.interex.org/pubcontent/enterprise/may01/14drudis.html [2002, MARCH, 5].

FINKELSTEIN, CLIVE, (1989)
"An Introduction To INFORMATION ENGINEERING- From Strategic Planning To Information Systems.": ADDISON-WESLEY PUBLISHING COMPANY.

FINKELSTEIN, CLIVE, (1993)
"Information Engineering: Strategic Systems Development.": ADDISON-WESLEY PUBLISHING COMPANY.

FISCHMAN, JOSHUA, & SEIFE, CHARLES, (1996)
"Working The Web With A Virtual Lab And Some Java.": SCIENCE, VOL. 273, NO. 5275, AUGUST, 1996.

FITZGERALD, BRIAN, (2000)
"Systems Development Methodologies: The Problems Of Tenses.": INFORMATION TECHNOLOGY.AND PEOPLE, VOL. 13, NO. 3, 2000.MCB UNIVERSITY PRESS.

FLYNN, DONAL, J., (1992)
"Information Systems Requirements: Determination And Analysis.": MCGRATH-HILL, LONDON.

FLYNN, DONAL, J., (1998)
"Information System Requirements: Determination And Analysis.":Second ed., MCGRAW-HILL, LONDON.

FOLKES, S., & STUBENVOLL, S., (1992)
"Accelerated Systems Development.": PRENTICE HALL, NEW YORK.

FOURNIER, ROGER, (1999)
"A Methodology for Client/Server And Web Application Development.": YOURDON PRESS, COMPUTING SERIES.

FRANKEL, SALLY, H., (1999)
"Web Applications: The Next Era In Computer Technology.": AVAILABLE ONLINE http://www.salnick.com/projects/WebApplications.htm [2002, MARCH, 14].

FRASINCAR, FLAVIUS, & JAN HOUBEN, GEERT, & VDOVJAK, RICHARD, (2001)
"An RMM-Based Methodology For Hypermedia Presentation Design": SPRINGER-VERLAG BERLIN HEIDELBERG, AVAILABLE ONLINE http://wwwis.win.tue.nl/~hera/papers/adbis2001/dm.pdf. [2003, JANUARY, 10].

FRATERNALI, P., & PAOLINI, P., (1998)
"A Conceptual Model And A Tool Environment For Development More Scalable And Dynamic Web Applications.":IN; H-J. SCHEK ET AL.(EDS). PROCEEDINGS OF THE SIXTH INTERNATIONAL CONFERENCE ON TEXTEDNING DATABSE TECHNOLOGY PP421-434. VALENCIA ,SPAIN, MARCH, 1998.

GAEDKE, M., & GRAEF, G., (2000)
"Development And Evolution Of Web-Applications Using The WebComposition Process Model". INTERNATIONAL WORKSHOP ON WEB ENGINEERING AT THE 9TH INTERNATIONAL WORLD-WIDE WEB CONFERENCE (WWW9), AMSTERDAM, THE NETHERLANDS, MAY 15, 2000.

GALLAGHER, JOAN, & COUGHLAN, SIOBHAN, (1997)
"Modern Office Technology And Practice." : GILL AND MACMILLAN.

GANE, CHRIS, & SARSON, TRISH, (1979)
"Structured Systems Analysis: Tools and Techniques.": PRENTICE HALL.

GANE, CHRIS, (1989)
"Rapid Systems Development Using Structured Techniques And Relational Technology." :PRENTIC HALL, LONDON, ENGLEWOOD CLIFFS.

GANE, CHRIS, (1990)
"Computer Aided Software Engineering: The Methodologies, The Products And The Future.": PRENTICE HALL, LONDON, INTERNATIONAL.

GARZOTTO, F., PAOLINI, P., & SCHWABE D., (1993)
"HDM: A Model-Based Approach To Hypertext Application Design.": ACM TRANSACTIONS OF INFORMATION SYSTEMS, VOL. 11, NO.1, PG1-26.

GELLERSEN, HANS-W., & GAEDKE, MARTIN, (1999)
"Object-Oriented Web Application Development.": IEEE INTERNET COMPUTING AVAILABLE ONLINE http://computerorg/internet/ [2003, JANUARY, 15].

GELLERSEN, HANS-WERNER, & WICKE, ROBERT , GAEDKE, MARTIN, (1997)
"WebComposition: An Object-Oriented Support System For The Web Engineering Lifecycle.": AVAILABLE ONLINE, http://decweb.ethz.ch/WWW6/Technical/Paper232/paper232.html [2003, MARCH, 14].

GINIGE, ATHULA, & MURUGESAN, SAN, (2001a)
"Web Engineering: An Introduction.": IEEE MULTIMEDIA, VOL. 8, NO. 1, JAN-MARCH, 2001. AVAILABLE ONLINE, http://computer.org/multimedia/mu2001/pdf/u1014.pdf. [2002, MARCH, 28].

GINIGE, ATHULA, & MURUGESAN, SAN, (2001b)
"The Essence Of Web Engineering: Managing The Diversity And Complexity Of Web Application Development.": AVAILABLE ONLINE, http://www-itec.uni-klu.ac.at/~harald/proseminar/web2.pdf [2002, MARCH, 28].

GORRY, G., ANTHONY, & MORTON, MICHAEL SCOTT, (1971)
"A Framework For Management Information Systems.": SLOAN MANAGEMENT REVIEW 13.

GRAHAM, IAN, (1993)
"Object-Oriented Methods.": ADDISON-WESLEY PUBLISHING COMPANY.

GREEN, STEVE, (1996)
"Information Systems Design.": INTERNATIONAL THOMPSON COMPUTER PRESS.

HACKATHORN, R.D., (1988)
"End User Computing By Top Executives.": DATABASE, VOL. 19, NO. 1, (1988).

HALASZ, F., & SCHWARTZ, M., (1990)
"The Dexter Reference Model." : PROCEEDINGS OF THE HYPERTEXT STANDARDISATION WORKSHOP PP. 95 – 133. GAITHERSBURY, MARYLAND, NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY PUBLICATION 500-178.

HASSAN, AHMED, E., & HOLT, RICHARD, C., (2001)
"Towards A Better Understanding Of Web Applications.": PROCEEDINGS OF WSE 2001. INTERNATIONAL WORKSHOP ON WEB SITE EVOLUTION, FLORENCE, ITALY, NOVEMBER 10, 2001.
AVAILABLE ONLINE.
http://www.plg.uwaterloo.ca/~aeehassa/home/pubs/wse2001.pdf [2003.JUNE.10].

HENDERSON-SELLERS, BRIAN, & EDWARDS, JULIAN M., (1990)
"The Object-Oriented Systems Life Cycle.": COMMUNICATIONS OF THE ACM, VOL. 33, NO.9, SEPTEMBER, 1990.

HENDERSON-SELLERS, BRIAN, (1992)
"A Book Of Object-Oriented Knowledge, Object Oriented Analysis, Design and Implementation: A New Approach to Software Engineering.": PRENTICE HALL.

HEREL, HEATH, H., (1999)
"*Productivity Applications." :PC MAGAZINE, OCTOBER 13, 1999.*

HIRSCHHEIM, R., & KLEIN, HEINZ, K., & LYYTINEN, KALLE, (1995)
"Information Systems Development And Data Modelling: Conceptual And Philosophical Foundations.": CAMBRIDGE UNIVERSITY PRESS.

HOFFER, JEFFREY, A., & GEORGE, JOEY, F., & VALACICH, JOSEPH, S., (2002)
"Modern System Analysis And Design." :Fifth ed., MCGRAW HILL.

ISAKOWITZ, TOMÁS, & BIEBER, MICHAEL, & VITALI, FABIO, (1998)
"Web Information Systems.": COMMUNICATIONS OF THE ACM VOL.,41, NO. 7 JULY 1998.

JACOBSON, IVAR, & CHRISTERSON, MAGNUS, & JONSSON, PATRIK, & ÖVERGAARD, GUNNER, (1993)
"Object-Oriented Software Engineering: A Use Case Driven Approach.": ADDISON-WESLEY PUBLISHING COMPANY.

KEEN, PETER, G., & SCOTT-MORTON, MICHAEL, S., (1978)
"Decision Support Systems: An Organisational Perspective.": READING MASS. ADDISION-WESELY PUBLISHING COMPANY.

KEEN, PETER, G.W., (1981)
"Value Analysis: Justifying Decision Support Systems.": MIS QUARTERLY, VOL. 5, NO.1, MARCH, 1981.

KENDALL, JULIE, E., & KENDALL, KENNETH, K., (1999)
"System Analysis And Design: Fourth Edition.": PRENTICE HALL.

KERR, JAMES, M., (1991)
"The Information Engineering Paradigm." : JOURNAL OF SYSTEMS MANAGEMENT, VOL. 42 NO. 4, PG 28, APRIL, 1991.

KOCH, NORA, (2001)
"A Comparative Study Of Methods For Hypermedia Development.": AVAILABLE ONLINE
http://www.dsic.upv.es/~west2001/iwwvst01/files/contributions/norakock/hypdev.pdf [2003, May, 20].

LANG, MICHAEL, (2001)
"A Study of Practice In Hypermedia Systems Design. ": AVAILABLE ONLINE, http://ecis2001.fov.uni-mb.si/doctoral/Students/ECIS-DC_Lang.pdf [2003, MAY, 22].

LAUDON, K.C., & LAUDON, J.P., (1995)
"Management Information Systems: A Contemporary Perspective.": MACMILLAN, BASINGSTOKE.

LEE, D.T., (1987)
"Computer Information System Development Methodologies: A Comparative Analysis.": AFIPS CONFERENCE PROCEEDINGS, 56, NATIONAL COMPUTER CONFERENCE, AFIPS PRESS, RESTON, VA.

LEE, HEESEOK, & KIM, JONGHO, KIM, YOUNG GUL, & CHO, SEON H. (1999a)
"A View Based Hypermedia Desing Methodologyy.": JOURNAL OF DATABASE MANAGEMENT. VOL. 10, NO. 2 APR-JUN 1999 PP 3-13.

LEE, HEESEOK, & LEE, CHOONGSEOK, & YOO, CHEONSOO (1999b)
"A Scenario-Based Object-Oriented Hypermedia Design Methodology.": INFORMATION & MANAGEMENT Vol. 36, PP121-138.

LEJK, M., & DEEKS, D., (1998)
"An Introduction To Systems Analysis Techniques.": PRENTICE HALL, EUROPE.

LEJK, M., & DEEKS, D., (2002)
"An Introduction To Systems Analysis Techniques.": Second ed., PRENTICE HALL, NEW YORK, HARLOW.

LEWIS, PAUL, J., (1994)
"Information-Systems Development: Systems Thinking In The Field Of Information-Systems.": PITMAN PUBLISHING.

LONGWORTH, GORDON, (1985)
"Designing Systems For Change." : *NCC PUBLICATIONS.*

LOWE, D., & HALL, W., (1999)
"Hypermedia & The Web: An Engineering Approach.":
JOHN WILEY.& SONS.

LOY, PATRICK, H., (1989)
"A Comparison Of Object-Oriented And Structured Development Methods.": A PAPER PRESENTED AT THE 1989 PACIFIC NORTHWEST CONFERENCE ON SOFTWARE QUALITY.

LU, MING-TE, & YEUNG, WING-LOK, (1998)
"A Framework For Effective Commercial Web Application Development.":
INTERNET RESEARCH, ELECTRONIC NETWORKING APPLICATIONS AND POLICY, VOL. 8, N0. 2, 1998. MCB UNIVERSITY PRESS.

LUCEY, TERRENCE, (1991)
"Management Information Systems.": D.P. PUBLISHERS, LONDON.
MACROMEDIA
AVAILABLE ONLINE, http://www.macromedia.com [2003,MAY, 16].

MADDISON, R.N., (1983)
"Information System Methodologies.": A WILEY HEYDEN PUBLICATION.

MARIE C., HOEPFL (1997)
"Choosing Qualitative Research: A Primer for Technology Education Researchers".:
JOURNAL OF TECHNOLOGY EDUCATION VOL. 9, NO. 1, 1997

MCCLURE, STEVE, (1998)
"*Web Application Development Developer Perspectives.": AN IDC WHITE PAPER, AVAILABLE ONLINE,*
http://www.macromedia.com/v1/documents/reports/idcreport.html [2002, APRIL, 24].

MCFADDEN, F.R., & HOFFER, J.A., & PRESCOTT, M.B., (1999)
"Modern Database Management." :Fifth ed., ADDISON-WESLEY.

MCLEOD, RAYMOND, (1993)
"Management Information Systems.": Fifth ed., MACMILLAN.

MICROSOFT CORPORATION
AVAILABLE ONLINE, www.microsoft.com [2002, DECEMBER, 10].

MONTLICK, TERRY, (2003)
"What Is Object-Orientated Software.": (Software Intelligence: Teach-In) SOFTWARE WORLD, VOL. 34 NO. 2, PG 3, MARCH, 2003.

MORGAN, GERARD, & O'NEILL, SEAMUS, (1996)
"Essential Computer Applications: Data-Bases, Spreadsheets And Word-Processing.":Second ed., GILL & MACMILLAN.

MORRIS, CHARLIE, (1999)
"Tools Of The Trade.": WEB DEVELOPERS JOURNAL. AVAILABLE ONLINE http://www.webdevelopersjournal.com/columns/tools.htm [2003,JUNE, 13].

MOYNIHAN, EDDIE, (1993)
"Business Management And Systems Analysis." : ALFRED WALLER LIMITED PUBLISHERS.

MUMFORD, ERIN, (1983)
"Designing Human Systems – The ETHICS method.": MANCHESTER BUSINESS SCHOOL, UK.

MURUGESAN, SAN & DESHPANDE, YOGESH, & HANSEN, STEVE, & GINIGE, ATHULA, (1999)
"Web Engineering: A New Discipline For Development Of Web-Based Systems." :PROCEEDINGS OF THE FIRST ICSE WORKSHOP ON WEB ENGINEERING, INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, LOS ANGELAS, MAY 1999. AVAILABLE ONLINE: http://aeims.uws.edu.au/WebEhome/ICSE99-WebE-Proc/San.doc [2002, MARCH, 22].

NANARD, J., & NANARD, M., (1995)
"Hypertext Design Environments And Hypertext Design Proccess.": COMMUNICATION OF THE ACM, AUGUST 1995.

NORMAN, R.J., (1996)
"Object-Oriented Systems Analysis And Design." :PRENTICE HALL INTERNATIONAL, LONDON.

O'BRIEN, JAMES A., (1991)
"Management Information Systems: Managing Information Technology In The International Enterprise." : Fourth ed., MCGRAW HILL.

O'BRIEN, JAMES A., (1999)
"Management Information Systems: Managing Information Technology In The Interworked Enterprise.": Fourth ed., IRWIN MCGRAW HILL

O'BRIEN, JAMES, A., (1998)
"Introduction To Information Systems: An Internetworked Enterprise Perspective.":Second ed., IRWIN/MCGRAW HILL.

OLLE, W.T., & HAGELSTEIN, J., & MACDONALD, I.G., & ROLLAND, C., & SOL, H.G., & VAN ASSCHE, F.J.M., & VERRIJN-STUART, A.A., (1991)
"Information Systems Methodologies; A Framework for Understanding.": Second ed., ADDISON-WESLEY.

OLSINA, L., (1998)
"Building A Web-Based Information System Applying The Hypermedia Flexible Process Modelling Strategy." 1ST INTERNATIONAL WORKSHOP ON HYPERMEDIA DEVELOPMENT, HYPERTEXT 98.

PARKER, C.S., (1998)
"Understanding Computers Today And Tomorrow." : 98 Edition, DRYDEN PRESS.

PARKER, CHARLES, & CASE, THOMAS, (1993)
"Management Information Systems: Strategy And Action." :Second ed., MCGRAW HILL.

PERFETTI, CHRISTINE, (2002)
"Flash Strikes Back: Creating Powerful Web Applications." :AVAILABLE ONLINE, http://www.uie.com/articlrs/flash_strikes_back/

PETERS, LAWRENCE, (1988)
"Advanced Structured Analysis And Design." :PRENTICE HALL, INTERNATIONAL EDITIONS.

PETTIT, STEVE., (2001)
"Anatomy of A Web Application: Security Considerations." [White Paper] SANCTUM INC, AVAILABLE ONLINE www.SanctumInc.com [2002, July, 19].

PHANOURIOU, CONSTANTINOS, (1996)
"Web Applications." :AVAILABLE ONLINE http://ei.sc.vt.edu/~wwwbtb/book/chap22/ [2002, MAY, 24]

PRESSMAN, ROGER, S., (2000)
"Software Engineering: A Practitioners Approach, European Adaptation.": Fifth ed.,MCGRAW-HILL PUBLISHING.

PRIESTLY, MARK, (2000)
"Practical Object-Oriented Design With UML. ":MCGRAW HILL.

ROBERTS-WITT, SARAH, L., (1999a)
"Human Resources.": PC MAGAZINE, OCTOBER, 13, 1999.

ROBERTS-WITT, SARAH, L., (1999b)
"Payroll & Accounting.": PC MAGAZINE, OCTOBER 29, 1999.

ROBERTS-WITT, SARAH, L., (1999c)
"Enterprise Resource Planning.": PC MAGAZINE OCTOBER, 13, 1999.

ROBERTS-WITT, SARAH, L., (1999d)
"Travel and Expenses.": PC MAGAZINE OCTOBER, 13,1999.

RODRIGUEZ-MEDINA, MANELI, & GIACHETTI, RONALD, (2001)
"An Investigation Of Design Methodologies For Internet Business Application Development.": PROCEEDINGS OF THE TWELFTH ANNUAL CONFERENCE OF THE PRODUCTION AND OPERATIONS MANAGEMENT SOCIETY, POM-2001.

ROSSI, GUSTAVO, & SCHWABE, DANIEL, & GUIMARAES, ROBSON, (2000)
"Designing Personalised Web Applications.": PROCEEDINGS OF THE 10TH WORLD WIDE WEB CONFERENCE, HONG KONG, CHINA, 2000.

ROSSI, GUSTAVO, & SCHWABE, DANIEL, & LYARDET, FERNANDO, (1999)
"Web Application Models Are More Than Conceptual Models. ": PROCEEDINGS OF THE WORLD WILD WEB AND CONCEPTUAL MODELLING'99 WORKSHOP, ER'99 CONFERENCE, SPRINGER, PARIS.

RUMBAUGH, J., BLAHA, M., PREMERLANI, W., & EDDY, F., & LORENSON, W., (1991)
"Object-Oriented Modeling And Design.": ENGLEWOOD CLIFFS, NJ. PRENTICE HALL.

RUSSO, NANCY, L., & FITZGERALD, BRIAN, & STOLTERMAN, ERIK, (2002)
"Information Systems Development: Methods In Action." : *MCGRAW-HILL.*

SATZINGER, JOHN, W., & JACKSON, ROBERT, B., & BURD, STEPHEN, D., (2002)
*"Systems Analysis And Design In A Changing World" Second ed.,*THOMSON LEARNING.

SCHWABE, DANIEL, & ROSSI, GUSTAVO, & ESMERALDO, LUISELENA, & LYARDET, FERNANDO, (2000)
"Web Design Frameworks: An Approach To Improve Reuse In Web Applications.": PROCEEDINGS OF THE SECOND INTERNATIONAL WORKSHOP ON WEB ENGINEERING, WWW9, CONFERENCE, SPRINGER, VERLAG, 2000.

SCHWABE, DANIEL, & DE ALMEIA PONTES, RITA (1998)
"OOHDM-Web:Rapid Prototyping Of Hypermedia Appliations In The Www.": MCC 08/09, MARCH 1998. DEPARTMENT OF INFOMATICS, PONTIFICIA UNIVERSIDADE CATOLICA DO RIO DE JANEIRO.

SCHWABE, DANIEL, & DE ALMEIDA PONTES, RITA, (1999a)
"A Method-Based Web Application Development Environment.": POSITION PAPER, WEB ENGINEERING WORKSHOP, WWW8.

SCHWABE, DANIEL, & DE ALMEIDA, PONTES, RITA, & MOURA, ISABELA, (1999b)
"OOHDM-Web: An Environment for Implementation of Hypermedia Applications in the WWW": SIGWEB NEWSLETTER, VOL. 8, NO. 2, JUNE, 1999.

SCHWABE, DANIEL, & ROSSI, GUSTAVO, & BARBOSA, SIMOME D.J., (1996)
"Systematic Hypermedia Application Design With OODHM.": PROCEEDINGS OF THE ACM INTERNATIONAL CONFERENCE ON HYPERTEXT, HYPERTEXT'96.

SCHWABE, DANIEL, & ROSSI, GUSTAVO, (1995)
"The Object-Oriented Hypermedia Design Model.": COMMUNICATIONS OF THE ACM, VOL. 38, NO. 8 PP45-48.

SEYMOUR, JIM, (1999)
"Web-Based Applications.": PC MAGAZINE, OCTOBER, 13,1999.

SHLAER, SALLY, & MELLOR, STEPHEN, J., (1988)
"Object-oriented Systems Analysis: Modelling the World in Data." YOURDON PRESS

SIMON, ERROL, (1996)
"Distributed Information Systems.": From Client/Server To Distributed Multimedia.":MCGRAW HILL.
SIMONE, LUISA, (1999a)
"Graphics Software.": PC MAGAZINE OCTOBER, 13, 1999.

SIMONE, LUISA, (1999b)
"Virtual Desktops.": PC MAGAZINE, OCTOBER, 13,1999.

SKIDMORE, STEVE, (1994)
"Introducing Systems Analysis.": BLACKWELL PUBLISHERS.

SOFTQUAD
AVAILABLE ONLINE
http://www.softquad.com

SULLY, PHIL, (1993)
"Modelling The World With Objects.": PRENTICE HALL.

TANENBAUM, ANDREW, S., & VAN STEEN, MAARTEN, (2002)
"Distributed Systems: Principles And Paradigms.": PRENTICE HALL.

THOMAS, DAVE, (1998)
"Web time Software Development." : SOFTWARE DEVELOPMENT MAGAZINE, OCTOBER 1998, PP78-80.

TOMÁS, ISAKOWITZ, & STOHR, EDWARD, A., & BALASUBRAMANIAN, P., (1995)
"RMM: A Methodology For Structured Hypermedia Design.": COMMUNICATIONS OF THE ACM, VOL. 38, NO. 8,PG 33-44 (1995).

TREPPER, CHARLES, (2000)
"Methodologies Vary According To The Project.": INFORMATION WEEK, AUGUST 21, 2000.

TRIBBLE, GUY, (1996)
"Java Computing In The Enterprise Revolution.": COMPUTER RESELLER NEWS, OCTOBER 21, 1996, NO. 706.

WALSH, IVAN, (2000)
"The Power Of Web Content Management.": TECHWATCH, AVAILABLE ONLINE http://www.techwatch.ie/fea/2000_513.htm [2002, JANUARY, 3].

WALTERS, S.A., & BROADY, J.E., & HARTLEY, R.J., (1994)
"A Review Of Information Systems Development Methodologies.": LIBRARY MANAGEMENT, VOL.15, NO. 6, MCB UNIVERSITY PRESS.

WEGNER, P., (1990)
"Concepts And Paradigms Of Object-Oriented Programming-Expansion of October 4. OOPSLA-89 Keynote Talk.": OOPS MESSENGER,1(1), ACM PRESS.

WHITE PAPER, (1998)
"An Overview of Java Technology.":
AVAILABLE ONLINE,
http://www.sun.com/access/artilces/JavaUniverseOverview.html

WHITE PAPER, (2000)
"COLDFUSION 4.5." : ALLAIRE CORPORATION, *AVAILABLE ONLINE*
http://www.allaire.com

WHITE PAPER, (2001)
"Functional Requirements And Use Cases.":
BREDEMEYER CONSULTING.

WHITE PAPER, (2002)
"The Challenges Of Building Distributed .NET Applications.":
AVAILABLE ONLINE,
http://www.compuware.com/whitepapers/resources/challenges.pdf

WHITTEN, JEFFREY, L., & BENTLY, LONNIE, D., & DITTMAN, KEVIN, C., (1998)
"Systems Analysis And Design Methods." :Fourth ed., IRWIN/MCGRAW HILL.

WHITTEN, JEFFREY, L., & BENTLY, LONNIE, D., & DITTMAN, KEVIN, C., (2001)
"Systems Analysis And Design Methods.": Fifth ed., IRWIN/MCGRAW HILL.

WILKIE, GEORGE, (1994)
"Object-Oriented Software Engineering: The Professional Developers Guide.":
ADDISON-WESLEY.

WOOD-HARPER, A.T., & ANTILL, LYN, & AVISON, D.E., (1990)
"Information Systems Definition: The Multiview Approach.": BLACKWELL SCIENTIFIC PUBLICATIONS.

YANG, JI-TZAY, & HAUNG, JIUN-LONG, & WANG, FENG-JIAN, & CHU, WILLIAM, C., (1999)
"Constructing Control-Flow-Based Testing Tools For Web Applications.":
PROCEEDINGS OF THE 11TH SOFTWARE ENGINEERING AND KNOWLEDGE ENGINEERING CONFERENCE (SEKE), JUNE 1999.

YEATES, DON, & SHIELDS, MAURA, & HELMP, DAVID, (1994)
"Systems Analysis And Design." :PITMAN , LONDON.

YOURDON, EDWARD, & ARGILA, CARL, (1996)
"Case Studies In Object Oriented Analysis And Design.": YOURDON PRESS COMPUTING SERIES 1996.

YOURDON, EDWARD, (1976)
"Managing The Structured Techniques.": Third ed., PRENTICE-HALL INTERNATIONAL, INC.

YOURDON, EDWARD, (1986)
"Managing The Structured Techniques.": Third ed.,. PRENTICE-HALL INTERNATIONAL, INC.

YOURDON, EDWARD, (1989)
"Modern Structured Analysis.": ENGLEWOOD CLIFFS, N.J. LONDON, PRENTICE-HALL INTERNATIONAL.

YOURDON, EDWARD, (1994)
"Object-Oriented Systems Design: An Integrated Approach.": PRENTICE HALL INTERNATIONAL EDITIONS.

APPENDIX A

Accompanying Letter and Definitions

Postgraduate Studies
Business Studies Department
Galway-Mayo Institute of Technology
Dublin Road
Tel: 753161 Ext. 2396
swalsh@gmit.ie

Dear Sir/Madam

As a postgraduate student in Business Studies, I am currently undertaking imperative research at the Galway-Mayo Institute of Technology (GMIT). The research focuses on the Methodologies, Techniques and Tools in the Development of Web Applications.

I would sincerely appreciate if you would take the time to meet me and answer my questions, as your expert opinions will be significant to the study. Without your help, the study will not be complete.

You can review the results of the survey before publication. Any concerns you have in relation to confidentiality will be thoroughly respected.

I appreciate that time is scarce but if you could find time next week, I would be extremely grateful. Please contact me at the above phone number or e-mail address to arrange a suitable time for such an interview.

Yours truly,

_____
Sylvia Walsh, BA.,

DEFINITIONS

System Types

The majority of questions require separate answers for two different system types: Web-based applications and Web sites. The following is a definition of each system type.

Web-Based applications: Applications that are based on a server that is attached to the Web and can be viewed and downloaded by anyone attached to the Web. Web-based applications can be applications that may be held on an internal server but can only be viewed using a Web browser.

Web sites: These are mostly static in nature and make up the majority of sites on the Web. Any Web site that has an application running behind it will be classed as a Web-based application.

APPENDIX B

Questionnaire

Questionnaire

(Web-Based Applications and Web sites)

Question 1 and question 2 may not be applicable.

[1] When did your company start trading?

Previous to 1990	
1990 – 1994	
1995 – 1999	
Later than 1999	

Please tick appropriate box

[2] Please indicate the number of employees in your organisation:

<50	51 – 100	101 – 500	>500

Please tick appropriate box

[3] Approximately into what bracket would you put the average cost of

(a) Web-based applications

(b) Web sites developed by you over the past five years.

	(a) Web-based applications	(b) Web Sites
Less than €5,000		
€5,000 - €19,000		
€20,000 - €49,000		
€49,000 - €99,000		
€100,000 - €500,000		
Greater than €500,000		

Please tick appropriate boxes

[4] With regard to the average project developed by you, please

(a) Specify the number of employees with some responsibility within each of the following areas and

(b) Estimate the percentage of your overall development effort expended within each of the same areas.

Functional Area	**No. of Employees**	**% Effort**
Project planning and management		
Systems analysis		
Systems design and coding		
Testing and debugging new information systems		
Modifying or enhancing systems previously developed		

Please insert appropriate numbers in each box

[5] What percentage of systems being developed by your organisation are predominantly:

(a) Web-based applications

(b) Web sites

Information System Type	**Percentage of Systems Developed**
Web-based applications	%
Web sites	%
Total	100%

Please insert the approximate percentages in each box

[6] Please indicate whether or not (a) Web-based applications and (b) Web sites are

I. Decreasing as a proportion of overall systems developed by you,

II. Remaining stable as a proportion of overall systems developed by you,

III. Increasing as a proportion of overall systems developed by you.

Systems Type	**(I) Decreasing**	**(II) Unchanging**	**(III) Increasing**
Web-based applications			
Web sites			

[7] With respect to **required functionality**, please rate the complexity of the average Web-based application developed by you on a scale of one to five.

Place X on appropriate point on line.

Not Complex 1.....|.....2.....|.....3.....|.....4.....|.....5 Extremely Complex

[8] With respect to **required functionality**, please rate the complexity of the average Web site developed by you on a scale of one to five.

Place X on appropriate point on line.

Not Complex 1.....|.....2.....|.....3.....|.....4.....|.....5 Extremely Complex

[9] With respect to **data structures**, please rate the complexity of the average Web-based application developed by you on a scale of one to five.

Place X on appropriate point on line.

Not Complex 1.....|.....2.....|.....3.....|.....4.....|.....5 Extremely Complex

[10] With respect to **data structures**, please rate the complexity of the average Web site developed by you on a scale of one to five.

Place X on appropriate point on line.

Not Complex 1.....|.....2.....|.....3.....|.....4.....|.....5 Extremely Complex

[11] With respect to **user-interfaces**, please rate the complexity of the average Web-based application developed by you on a scale of one to five.

Place X on appropriate point on line.

Not Complex 1.....|.....2.....|.....3.....|.....4.....|.....5 Extremely Complex

[12] With respect to **user interfaces**, please rate the complexity of the average Web site developed by you on a scale of one to five.

Place X on appropriate point on line.

Not Complex 1.....|.....2.....|.....3.....|.....4.....|.....5 Extremely Complex

[13] If user interface design is very complex, please identify the reasons why?

[14] How do you deal effectively with user interface design complexity?

[15] From the list of methodologies below, please indicate the methodologies (if any),

(a) **currently** used by you as your principal development methodology, and

(b) **previously** used by you as your principal development methodology.

Development Methodology	**Web- Based Applications**		**Web Sites**	
	Currently Used	***Previously Used***	***Currently Used***	***Previously Used***
Dexter Hypertext Reference Model				
EORM				
eW3DT				
HDM				
HDM2				
HDM-Lite				
HFPM				
Lowe-Hall's Engineering Approach				
MacWeb Approach				
OMT				
OO/Pattern Approach				
OOHDM				
OOHDM-Web				
RMM				
RNA				
SOHDM				
UML				
Usage Centred Design				
VHDM				
Web Engineering				
WebComposition				
WSDM				
Other Please Specify				

Please tick appropriate boxes for each system type

[16] What are the strengths of this particular methodology?

[17] What are the weaknesses of this particular methodology?

[18] From the list of development tools or languages below, please indicate (if any),

(a) **currently** used by you as your primary software development tool, and
(b) **previously** used by you as your primary software development tool.

Software Development Tool	**Web- Based Applications**		**Web Sites**	
	Currently Used	***Previously Used***	***Currently Used***	***Previously Used***
Access				
Active X Components				
ASP				
C				
C++				
CGI				
Coldfusion				
CSS				
DHTML				
Dreamweaver				
Flash				
Frontpage				
GoLive				
Hotmetal				
HTML				
Java				
Java Applets				
Java Script				
Java Servlets				
JSP				
MS SQL				
MY SQL				
Oracle				
Perl				
Shockwave				
Sybase				
Ultra Dev				
Visual Basic				
Visual Basic Script				
Visual Net				
W3C's HTML Validator				
XML				
Other Please Specify				

Please tick the appropriate boxes for each system type

[19] What are the strengths of this particular tool?

[20] What are the weaknesses of this particular tool?

[21] From the list of structured techniques below, please indicate (if any),

(a) **currently** used by you in the development of Web-based application and Web sites, and

(b) **previously** used by you in the development of Web-based applications and Web sites.

Techniques	**Web- Based Applications**		**Web Sites**	
	Currently Used	***Previously Used***	***Currently Used***	***Previously Used***
Action Diagrams				
Data Flow Diagram				
Decision Tables				
Decision Trees				
Entity Life Histories				
Entity Relationship Diagram				
Normalisation				
Object-Oriented Techniques				
Program Flowcharts				
Pseudocode				
Scenarios				
Structured English				
System Flowcharts				
Warnier-Orr Diagrams				
UML				
Other Please Specify				

Please tick the appropriate boxes for each system type

[22] If you are planning to adopt specific new methodologies in the near future, please identify from the list below, and indicate whether you intend to adopt use of the specified methodology/ies

(a) within **one** year, or
(b) within **three** years.

Development Methodology	**Web- Based Applications**		**Web Sites**	
	Within One Year	***Within Three Years***	***Within One Year***	***Within Three Years***
Dexter Hypertext Reference Model				
EORM				
HDM				
HDM2				
HDM-Lite				
HFPM				
Lowe-Hall's Engineering Approach				
MacWeb Approach				
OMT				
OO/Pattern Approach				
OOHDM				
OOHDM-Web				
RMM				
RNA				
SOHDM				
UML				
Usage Centred Design				
VHDM				
W3DT				
Web Engineering				
WebComposition				
WSDM				
Other Please Specify				

Please tick the appropriate boxes for each system type

[23] If adopting a new methodology, what will be the deciding factor?

[24] If you are planning to adopt use of specific software development tools as your primary development tool in the near future, please identify from the list below, and indicate whether you intend to adopt use of each specified tool,

(a) within **one** year, or
(b) within **three** years.

Software Development Tool	**Web- Based Applications**		**Web Sites**	
	Within One Year	***Within Three Years***	***Within One Year***	***Within Three Years***
Access				
Active X Components				
ASP				
C				
C++				
CGI				
Coldfusion				
CSS				
Dreamweaver				
DHTML				
Flash				
Frontpage				
GoLive				
Hotmetal				
HTML				
Java				
Java Applets				
Java Script				
Java Servlets				
JSP				
MS SQL				
MY SQL				
Oracle				
Perl				
Shockwave				
Sybase				
Ultra Dev				
Visual Basic				
Visual Basic Script				
W3C's HTML Validator				
XML				
Visual Net				
Other Please Specify				

Please tick the appropriate boxes for each system type

[25] If adopting a new development tool, what will be the deciding factor?

[26] If you are planning to adopt use of new structured technique in the near future, please identify from the list below, and indicate whether you intend to adopt use of each structured technique,

(a) within **one** year, or
(b) within **three** years.

Techniques	**Web- Based Applications**		**Web Sites**	
	Within one Year	***Within Three Years***	***Within One Year***	***Within Three Years***
Action Diagrams				
Data Flow Diagram				
Decision Tables				
Decision Trees				
Entity Life Histories				
Entity Relationship Diagram				
Normalisation				
Object-Oriented Techniques				
Program Flowcharts				
Pseudocode				
Scenarios				
Structured English				
System Flowcharts				
Warnier-Orr Diagrams				
UML				
Other Please Specify				

Please tick the appropriate boxes for each system type

[28] Do you conform to any of the following standards and guidelines?

Guideline	Web-based Applications	Web Sites
W3C's Web Content Accessibility Guidelines		
Web-Based Enterprise Management (WBEM)		
Common Information Model (CIM) Standards		
DMI Standards		
Directory Enables Network (DEN) Initiative		
Other Please Specify		

Please tick appropriate box for each system type

[27] Can I call you if necessary for clarification?